description of the residue code). If Address Search Mode is enabled, the receiver once again enters Address Search Phase.

Unlike the COP mode of operation, data from the Sync Comparison/Zero Deletion logic passes directly to the CRC checker. As a result, when the End of Frame Flag is detected, the CRC calculation is complete and the error status is passed to the status buffer along with the residue code. The CRC checker is automatically reset to all ones at this time.

G-4.3 BUS INTERFACE CONTROLLER

The bus interface controller is the interface between the transmitter and receiver sections and the processor bus. The major components of this section are shown in Figure G-4.5. The control and status registers pertinent to the operation of the control section are illustrated in Table G-4.4.

The bus interface controller can be divided into four major components:

Bus Control Logic
Interrupt Control Logic
DMA Control Logic
Clock and Reset Control Logic

All of these components interact to provide a flexible high-performance interface between the bus architecture defined by your processor and application and the various internal elements that make up the MPSC.[2]

G-4.3.1 Bus Control Logic

The bus control logic determines the direction and internal source or destination of data and control transfers between the MPSC[2] and the processor bus. During operation of the MPSC[2], the bus control logic may operate in any of three distinct modes: Processor Read/Write, Interrupt Acknowledge, and DMA Cycle. These last two modes are described in detail in Sections G-4.3.2 and G-4.3.3.

Processor Read/Write mode is the normal mode of operation. The processor transfers data or commands and status to or from the MPSC[2] with its instruction set. The MPSC[2] is enabled for Processor Read/Write mode when the chip select (CS) input is made active (low). The direction of the transfer is controlled by enabling either the read (RD) or write (WR) inputs. The B/A input determines the source/destination channel for the transfer and the C/D input specifies whether the transfer is character data or control/status information. These inputs are generally connected to the two low-order address lines. Figure 6.1 illustrates a typical connection between a processor and the MPSC[2].

Table G-4.3 Read/Write Selection

$\overline{CS}$	B/$\overline{A}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	OPERATION
1	X	X	X	X	NO OPERATION. THE MPSC² IS DESELECTED.
0	X	X	1	1	NO OPERATION. THE MPSC² IS DESELECTED.
0	0	0	1	0	WRITE A CHAR TO CHANNEL A TRANSMITTER.
0	0	0	0	1	READ A CHAR FROM CHANNEL A RECIVER.
0	0	1	1	0	WRITE A CONTROL BYTE TO CHANNEL A.
0	0	1	0	1	READ A STATUS BYTE FROM CHANNEL A.
0	1	0	1	0	WRITE A CHAR TO CHANNEL B TRANSMITTER.
0	1	0	0	1	READ A CHAR FROM CHANNEL B RECEIVER.
0	1	1	1	0	WRITE A CONTROL BYTE TO CHANNEL B.
0	1	1	0	1	READ A STATUS BYTE FROM CHANNEL B.
0	X	X	0	0	ILLEGAL.

G-4.3.2 Interrupt Control Logic

The interrupt control logic performs two functions: it prioritizes various internal input requests, and places the appropriate information on the data bus during an Interrupt Acknowledge cycle (if you enabled the MPSC²s vectored interrupt feature).

Figure G-4.5 Bus Interface Controller

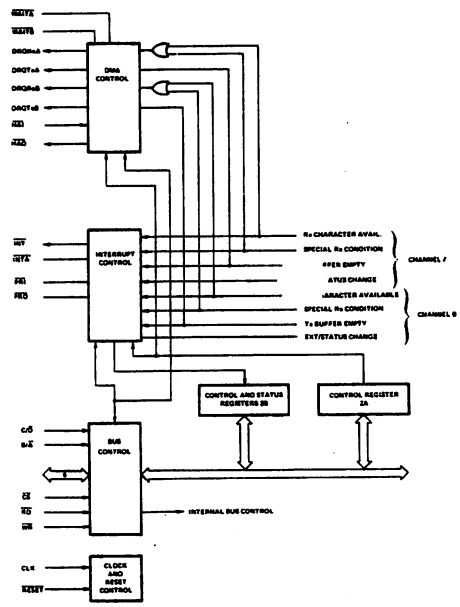


Table G-4.4 Bus Interface Controller Control and Status Registers

CONTROL REGISTER	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	*Relevant commands
CR0			COMMAND*			REGISTER POINTER			Channel Reset
CR2A		0	Vector Mode Select			Priority	DMA Mode Select		End of Interrupt
CR2B	INTERRUPT VECTOR								

STATUS REGISTER	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
CR0							Interrupt Pending	
CR2B	INTERRUPT VECTOR							

Each MSPC² channel can generate four different types of interrupt requests:

Received Character Available

Special Received Condition (character received but with an error or SDLC End of Frame flag received)

Transmitter Buffer Empty

External input (CTS, DCD, SYNC, Internal Status (Sync,Idle/CRC Latch) Change)

When any of these requests occurs, the interrupt control logic determines whether to accept the request at that time, issue an interrupt request by setting the INT output low when the request is accepted, and, if Vectored Interrupt mode is enabled, place the interrupt information on the data bus during the times that the interrupt acknowledge input (INTA) is activated by the processor.

As an example, assume that the channel A DCD input has just changed state causing an External/Status Change interrupt request. The following sequence occurs:

If all the following conditions are true:

External/Status Change interrupts are enabled

No higher priority interrupt requests are pending

PRI is active

The MPSC² is not acknowledging a pending lower priority interrupt request

Then the interrupt control logic accepts the interrupt request and sets INT active and PRO inactive.

If Vectored Interrupt mode is enabled, the MPSC² may place information on the data bus in response to a series of INTA pulses as shown in the following chart.

Table G-4.5 Vectored Interrupt Mode

Interrupt Mode Select	PRI	INTA Cycle		
		1	2	3
8080/5 Master	0	CD HEX (CALL OP)	VECTOR	0
	1	CD HEX (CALL OP)	HI-Z	HI-Z
8080/5 Slave	0	HI-Z	VECTOR	0
	1	HI-Z	HI-Z	HI-Z
8086	0	HI-Z	VECTOR	*
	1	HI-Z	HI-Z	*

*The 8086 issues 2 Interrupt Acknowledge pulses rather than 3.

When operating in the 8080/5 modes, the MPSC² issues an 8080-type CALL CD vv Hex instruction where vv is the contents of control register 2B (modified by the cause of the interrupt if the Status Affects Vector feature is enabled). In particular, an MPSC² programmed for 8085 Master mode always places the CALL opcode on the data bus regardless of whether that MPSC² has a pending interrupt request. To avoid problems caused by momentary bus contention, you should never program more than one device to operate in this mode.

In 8086 mode, the MPSC² places the vector on the data bus during the second interrupt acknowledge to vector the processor to the approximate location in low memory.

Figure G-4.6 MPSC² Interrupt Conditions

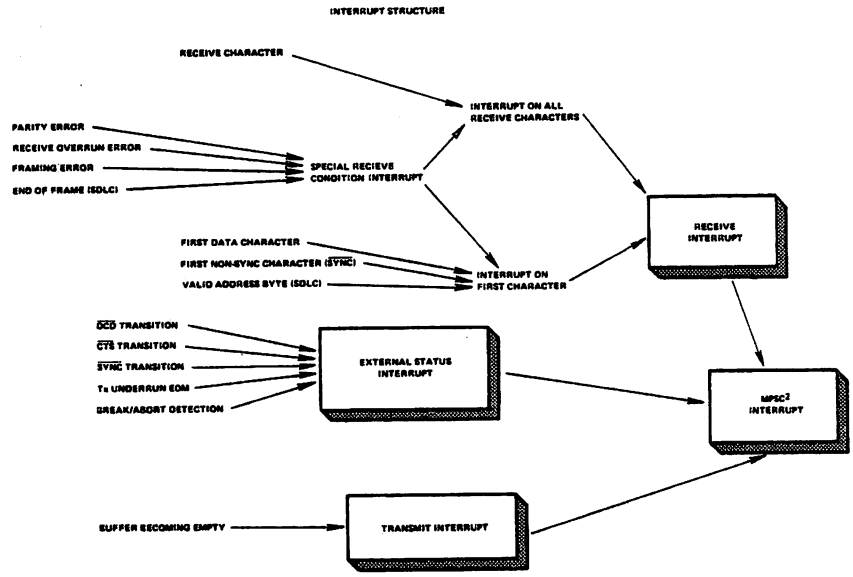
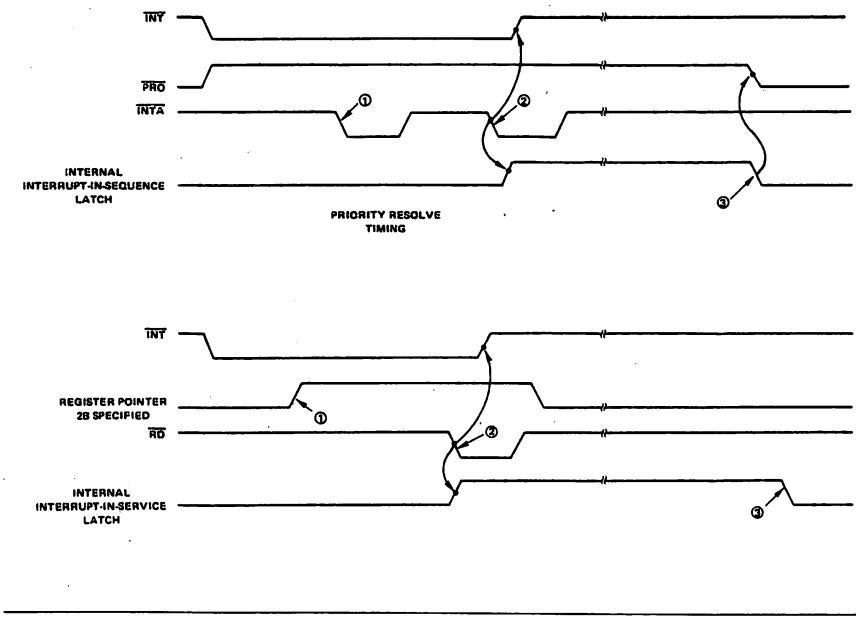


Figure G-4.7 illustrates the action of the interrupt control logic during an interrupt acknowledge sequence.

Figure G-4.7 Interrupt Timing



At the beginning of the first Interrupt Acknowledge cycle, the interrupt prioritization logic is frozen to permit any late interrupt requests by higher priority devices to ripple through and resolve internal priorities before the second interrupt pulse.

At the end of the second INTA pulse, the INT output is released by the acknowledging device and the interrupt prioritization logic is re-enabled with an Interrupt In Service flag set. As long as this flag is set, PRO is held high and only internal interrupt requests with a priority higher than the one currently being serviced are accepted.

While the interrupt is being serviced, the processor issues an End of Interrupt (EOI) command to the MPSC² to reset the interrupt control logic to its previous state. This scheme permits nested interrupts to be serviced and the priority daisy chain to be properly maintained.

When the MPSC² is operated in Non-vectored Interrupt mode, the interrupt control logic operates in a similar manner except that INTA is not used and no vector information is placed on the data bus. Rather, the interrupt acknowledge sequence is simulated by reading the vector (modified if Status Affects Vector is enabled) in status register 2B.

G-4.3.3 DMA Control Logic

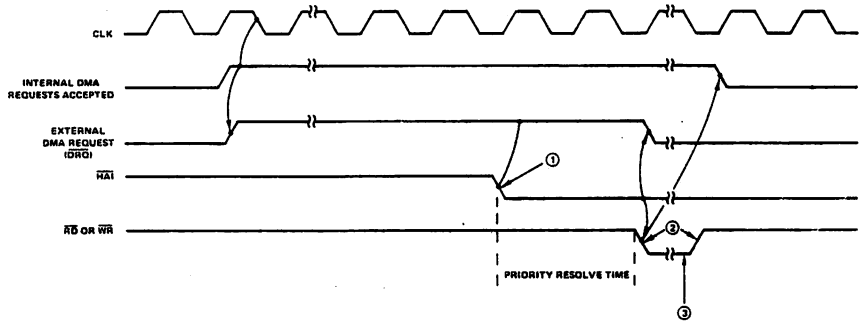
The function of the DMA logic is somewhat similar to that of the interrupt control logic in that service requests must be accepted, prioritized, and information placed on (or, in this case, accepted from as well) the data bus at the appropriate times. However, the purpose of the DMA control logic is to enable the MPSC² to avoid interrupting the processor to make a data transfer. This is accomplished by activating an external controller to move the data directly from the MPSC² to memory, or vice versa.

The DMA control logic accepts requests from four sources: (1) Received Data Available in channel A, (2) Transmitter Buffer Becoming Empty in channel A., (3) Data Available in channel B, and (4) Transmitter Buffer Becoming Empty in channel B. When an internal DMA request is made by one of the above sources and DMA mode is enabled for that channel, the appropriate DMA request output (e.g. DRQRxA when received data is available in channel A) is made active. This causes the external DMA controller to request control of the processor bus with a hold request. The MPSC²'s daisy chain output, HAO, is at this point locked in the inactive (high) state.

Some time later, the external DMA controller gains control of the processor bus as the processor asserts its hold acknowledge output.

The DMA Controller now places the source or destination address on the address bus and asserts the I/O read or write control line for a data transfer from or to the MPSC², respectively. The MPSC² also receives the processor hold acknowledge signal possibly through higher priority MPSC²s not requesting DMA, at its HAI input. When HAI is asserted, the DMA control logic freezes all internal requests, determines which one has the highest priority, and performs the transfer when I/O read or write is received from the DMA controller at RD or WR. Once the transfer is complete, the prioritization logic is re-enabled and new or pending requests can be serviced. Figure G-4.8 illustrates some of the timing details of a DMA transfer.

Figure G-4.8 DMA Data Transfer Timing

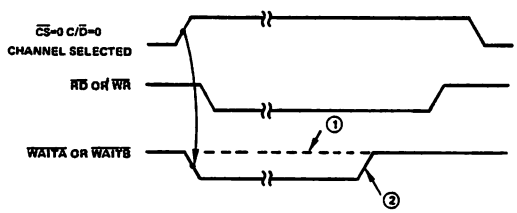


From the above explanation you should note two points. First, in the case of multiple DMA requests from one MPSC², both the MPSC² and the external DMA controller establish priorities independently to determine which request to service first. As a result, you MUST connect the MPSC²'s DMA request outputs to the DMA controller so that both make the same priority decisions. For example, when using the MPSC² with an 8257-type DMA controller and the priority bit ($CR2A\text{-}D_2$) = 0, you must set the controller to the fixed priority mode (as opposed to rotating priority), and connect the MPSC²s DRQRxA output to the 8257's DRQ 0 input, DRQTxA to DRQ 1, and so on.

The second point is that many DMA controllers, such as the 8257, may begin the transfer by asserting RD or WR before the MPSC² can receive HAI through the daisy chain and resolve request priorities. Because of this, you should always derive HLDA to the DMA Controller from HAI of the MPSC²(s) to which it is connected. Additionally, a delay circuit from HAI to HLDA is recommended. Figure G-6.5 shows a typical MPSC²/DMA interface which conforms to these points.

The mechanism that controls the WAIT outputs of the MPSC² is related to the DMA logic. When enabled, the wait logic pulls the WAIT line active when the processor attempts to perform a data transfer operation at an inappropriate time. If WAIT is connected to the processor's WAIT (or READY) input, it waits until the line is released by the MPSC² before completing the data transfer. Since the processor is dedicated to either a read or write operation at any one time, only one WAIT output is required for each channel. You may assign it to operate with either the transmitter or the receiver. Figure G-4.9 illustrates the basic wait feature timing.

Figure G-4.9 Wait Mode Timing



G-4.3.4 Clock and Reset Control Logic

The clock input of the MPSC² controls the various timing states of the MPSC² and is usually connected to the processor clock. The clock is not used by the bus control logic and data transfers need not be synchronized to it in any way. The receiver and transmitter sections use the clock, and it must be at least 4.5x the highest data clock frequency you plan to use. The DMA control logic also uses the clock, and it should be the same clock seen by the external DMA Controller.

The RESET input is used at power-up and at any other time that you wish to reset the MPSC² to its initial state. After a reset, all transmitters and receivers are disabled, any pending interrupt and DMA requests are cleared, and the modem control outputs DTR and RTS are reset (high). When you reset the MPSC², you must hold the RESET input low for at least one complete clock cycle.

G-5 PROGRAMMING THE MPSC²

The software operation of the MPSC² is very straightforward. Its consistent register organization and high-level command structure help to minimize the number of operations required to implement complex protocol designs. Programming is further simplified by the MPSC²s extensive interrupt and status reporting capabilities.

This section is divided into two parts. The first is a detailed description of the commands, bits, and fields in the various MPSC² control and status registers. The second part provides programming examples and flowcharts for the MPSC²'s various operating modes to assist you in developing software for your specific application.

G-5.1 THE MPSC² REGISTERS

The MPSC² interfaces to the system software with a number of control and status registers associated with each channel. Commonly used commands and status bits are accessed directly through control and status registers 0. Other functions are accessed indirectly with a register pointer to minimize the address space that must be dedicated to the MPSC².

Table G-5.1 Control Registers

CONTROL REGISTER	FUNCTION
0	FREQUENTLY USED COMMANDS AND REGISTER POINTER CONTROL
1	INTERRUPT CONTROL
2	PROCESSOR/BUS INTERFACE CONTROL
3	RECEIVER CONTROL
4	MODE CONTROL
5	TRANSMITTER CONTROL
6	SYNC/ADDRESS CHARACTER
7	SYNC CHARACTER

Table G-5.2 Status Registers

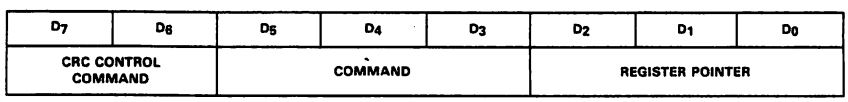
STATUS REGISTER	FUNCTION
0	BUFFER AND "EXTERNAL/STATUS" STATUS
1	RECEIVED CHARACTER ERROR AND SPECIAL CONDITION STATUS
2 (CHANNEL B ONLY)	INTERRUPT VECTOR

All control and status registers except CR2 are separately maintained for each channel. Control and status registers 2 are linked with the overall operation of the MPSC² and have different meanings when addressed through different channels.

When initializing the MPSC², control register 2A (and 2B if desired) should be programmed first to establish the MPSC² processor/bus interface mode. You may then program each channel to be used separately, beginning with control register 4 to set the protocol mode for that channel. The remaining registers may then be programmed in any order.

G-5.1.1 Control Register 0

Figure G- 5.1 Control Register 0



Register Pointer (D_0-D_2)

The register pointer specifies which register number is accessed at the next Control Register Write or Status Register Read. After a hardware or software reset, the register pointer is set to 0. Therefore, the first control byte goes to control register 0. When the register pointer is set to a value other than 0, the next control or status (C/D = 1) access is to the specified register, after which the pointer is reset to 0. You can freely combine other commands in control register 0 with setting the register pointer.

Command (D_3-D_5)

Commands commonly used during the operation of the MPSC² are grouped in control register 0. They are:

Null (000)

This command has no effect and is used when you wish to set only the register pointer or issue a CRC command.

Send Abort (001)

When operating in SDLC mode, this command causes the MPSC² to transmit the SDLC abort code, issuing 8 to 13 consecutive ones. Any data currently in the transmitter or the transmitter buffer is destroyed. After sending the abort, the transmitter reverts to the Idle Phase (flags).

Reset External/Status Interrupts (010)

When the External/Status Change flag is set, the condition bits D_0-D_2 of status register 0 are latched to allow you to capture short pulses that may occur. The Reset External/Status Interrupts Command clears a pending interrupt and re-enables the latches so that new interrupts may be sensed.

Channel Reset (011)

This command has the same effect on a single channel as an external reset at pin 2. A channel reset command to channel A resets the internal interrupt prioritization logic. This does not occur when you issue a Channel Reset command to channel B. You must reinitialize all control registers associated with the channel that you reset. After a channel reset, you must wait at least four system clock cycles before writing new commands or controls to that channel.

Enable Interrupt on Next Character (100)

When operating the MPSC² in Interrupt on First Received Character mode, you may issue this command at any time (generally at the end of a message), to re-enable the interrupt logic for the next received character.

Reset Pending Transmitter Interrupt/DMA Request (101)

You can reset a pending Transmitter Buffer Becoming Empty interrupt or DMA request without sending another character by issuing this command (typically at the end of a message). A new Transmitter Buffer Becoming Empty interrupt or DMA request is not made until another character has been loaded and transferred to the transmitter shift register or when, if operating in synchronous or SDLC mode, the CRC character has been completely sent and the first sync or flag character loaded into the transmitter shift register.

Error Reset (110)

This command resets a Special Receive Condition interrupt. It also re-enables the Parity and Overrun Error latches that allow you to

check for these errors at the end of a message.

End of Interrupt (111)(Channel A only)

Once an interrupt request has been issued by the MPSC², all lower priority internal and external interrupts in the daisy chain are held off to permit the current interrupt to be serviced while allowing higher priority interrupts to occur. At some point in your interrupt service routine (generally at the end), you must issue the End of Interrupt command to channel A to re-enable the daisy chain and allow any pending lower priority internal interrupt requests to occur.

CRC Control Commands (D_6-D_7)

These commands control the operation of the CRC generator/ checker logic.

Null (00)

This command has no effect and is used when issuing other commands or setting the register pointer.

Reset Receiver CRC Checker (01)

This command resets the CRC checker to 0 when the channel is in a synchronous mode and resets to all ones when in SDLC mode.

Reset Transmitter CRC Generator (10)

This command resets the CRC generator to 0 when the channel is in a synchronous mode and resets to all ones when in SDLC mode.

Reset Idle/CRC Latch (11)

This command resets the Idle/CRC latch so that when a transmitter underrun condition occurs (that is, the transmitter has no more characters to send), the transmitter enters the CRC Phase of operation and begins to send the 16-bit CRC character calculated up to that point. The latch is then set so that if the underrun condition persists, idle characters are sent following the CRC. After a hardware or software reset, the latch is in the set state.

G-5.1.2 Control Register 1

Figure G-5.2 Control Register 1

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
WAIT FUNCTION ENABLE	∅	WAIT ON RECEIVER TRANSMITTER	RECEIVER INTERRUPT MODE		CONDITION AFFECTS VECTOR	TRANSMITTER INTERRUPT ENABLE	EXT/STATUS INT ENABLE

External/Status Interrupt Enable (D_0)

When this bit is set to one, the MPSC² issues an interrupt whenever any of the following occur:

transition of DCD input

transition of CTS input

transition of SYNC input

entering or leaving synchronous Hunt Phase break detection or termination

SDLC abort detection or termination

Idle/CRC latch becoming set (CRC being sent)

Transmitter Interrupt Enable (D_1)

When this bit is set to one, the MPSC² issues an interrupt when:

the character currently in the transmitter buffer is transferred to the shift register (Transmitter Buffer Becoming Empty) or,

the transmitter enters Idle Phase and begins transmitting sync or flag characters.

Status Affects Vector (D_2)

When this bit is set to 0, the fixed vector programmed in CR2B during MPSC² initialization is returned in an interrupt acknowledge sequence. When this bit is set to 1, the vector is modified to reflect the condition that caused the interrupt. See Section G-5.1.12 for a detailed explanation of the MPSC²'s vectored interrupt feature.

Receiver Interrupt Mode (D_3-D_4)

This field controls how the MPSC²'s interrupt/DMA logic handles the character received condition.

Receiver Interrupts/DMA Request Disabled (00)

The MPSC² does not issue an interrupt or a DMA request when a character has been received.

Interrupt on First Received Character Only (01)

(and issue a DMA Request)

In this mode, the MPSC² issues an interrupt only for the first character received after an Enable Interrupt on First Character Command (CRO) has been given. If the channel is in DMA mode, a DMA request is issued for each character received including the first. This mode is generally used when using the MPSC² in DMA or Block Transfer mode to signal the processor that the beginning of an incoming message has been received.

Interrupt (and issue a DMA Request) (10)

On All Received Characters
Parity Error is a Special Receive Condition

In this mode, an interrupt (and DMA request if DMA mode is selected) is issued whenever there is a character present in the receiver buffer. A parity error is considered a special receive condition.

Interrupt (and issue a DMA request) (11)

On All Received Characters
Parity Error is not a Special Receive Condition

This mode is the same as above except that a parity error is not considered a special receive condition. The following are considered special receive conditions and, when status affects vector is enabled, cause an interrupt vector different from that caused by a received character available condition:

Receiver Overrun Error

Parity Error (if specified)

SDLC End of Message (final flag received)

Wait on Receiver/Transmitter (D_5)

If the Wait function is enabled for block mode transfers, setting this bit to 0 causes the MPSC² to issue a wait (WAIT output goes low) when the processor attempts to write a character to the transmitter while the transmitter buffer is full. Setting this bit to 1 causes the MPSC² to issue a wait when the processor attempts to read a character from the receiver while the receiver buffer is empty.

Wait Function Enable (D_7)

Setting this bit to 1 enables the wait function as described above and in Section 4.3.3.

G-5.1.3 Control Register 2 (Channel A)

Figure G-5.3 Control Register 2 (Channel A)

D7	D6	D5	D4	D3	D2	D1	D0
PIN 10 SYNCB/RTSB	0	INTERRUPT VECTOR MODE			PRIORITY	DMA MODE SELECT	

DMA Mode Select (D_0-D_1)

Setting this field establishes whether channels A and B are used in DMA mode (i.e. data transfers are performed by a DMA controller) or in non-DMA mode where transfers are performed by the processor in either Polled, Interrupt, or Block Transfer modes. The functions of some MPSC² pins are also controlled by this field.

Table G-5.3 DMA Mode Selection

		Channel		Pin Function					
D_1	D_0	A	B	11	26	29	30	31	32
0	0	Non-DMA	Non-DMA	$\overline{WAITB}$	$\overline{DTRB}$	$\overline{PRI}$	$\overline{PRO}$	$\overline{DTRA}$	$\overline{WAITA}$
0	1	DMA	Non-DMA	DRQTxA	$\overline{HAI}$	$\overline{PRI}$	$\overline{PRO}$	$\overline{HAO}$	DRQRxA
1	0	DMA	DMA	DRQTxA	$\overline{HAI}$	DRQRxB	DRQTxB	$\overline{HAO}$	DRQRxA
1	1	Illegal	–	–	–	–	–	–	–

Priority (D_2)

This bit allows you to select the relative priorities of the various interrupt and DMA conditions according to your application.

Table G-5.4 DMA/Interrupt Priorities

	Mode		DMA Priority Relation	Interrupt Priority Relation
D_2	CHA	CHB		
0	INT	INT	——	RxA > TxA > RxB > TxB > ExTA > ExTB
1			——	RxA > RxB > TxA > TxB > ExTA > ExTB
0	DMA	INT	RxA TxA	RxA > RxB > TxB > ExTA > ExTB
1			RxA TxA	RxA > RxB > TxB > ExTA > ExTB
0	DMA	DMA	RxA TxA RxB TxB	RxA > RxB > ExTA > ExTB
1			RxA RxB TxA TxB	RxA > RxB > ExTA > ExTB

Interrupt Vector Mode (D_3-D_5)

This field determines how the MPSC² responds to an interrupt acknowledge sequence from the processor. See Section 4.3.2 for a detailed description of the MPSC² response in these modes.

Table G-5.5 Interrupt Acknowledge Sequence Response

D_5	D_4	D_3	Mode	Status Register 2B and Interrupt Vector bits affected when Condition Affects Vector is enabled
0	0	0	Non-Vectored	D_4 D_3 D_2
0	0	1	Non-Vectored	D_4 D_3 D_2
0	1	0	Non-Vectored	D_2 D_1 D_0
0	1	1	Illegal	–
1	0	0	8085 Master	D_4 D_3 D_2
1	0	1	8085 Slave	D_4 D_3 D_2
1	1	0	8086	D_2 D_1 D_0
1	1	1	Illegal	–

Pin 10 SYNCB/RTSB Select (D_7)

Programming a 0 into this bit selects RTSB as the function of pin 10. A one selects SYNCB as the function.

G-5.1.4 Control Register 2 (Channel B)

Figure G-5.4 Control Register 2 (Channel B)

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
INTERRUPT VECTOR							

G-Interrupt Vector (D_0-D_7)

When the MPSC² is used in Vectored Interrupt mode, the contents of this register are placed on the bus during the appropriate portion of the interrupt acknowledge sequence. Its value is modified if status affects vector is enabled. You can read the value of CR2B at any time. This feature is particularly useful in determining the cause of an interrupt when using the MPSC² in Non-vectored Interrupt mode.

G-5.1.5 Control Register 3

Figure G-5.5 Control Register 3

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
NUMBER OF RECEIVED BITS/CHARACTER		AUTO ENABLES	ENTER HUNT PHASE	RECEIVER CRC ENABLE	ADDRESS SEARCH MODE	SYNC CHARACTER LOAD INHIBIT	RECEIVER ENABLE

Receiver Enable (D_0)

After the channel has been completely initialized, setting this bit to 1 allows the receiver to begin operation. You may set this bit to 0 at any time to disable the receiver.

Sync Character Load Inhibit (D_1)

In a synchronous mode, this bit inhibits the transfer of sync characters to the receiver buffer, thus performing a "sync stripping" operation. When using the MPSC²'s CRC checking ability, you should use this feature only to strip leading sync characters preceding a message since the load inhibit does not exclude sync characters embedded in the message from the CRC calculation. Synchronous protocols using other types of block checking such as checksum or LRC are free to strip embedded sync characters with this bit.

Address Search Mode (D_2)

In SDLC Mode, setting this bit places the MPSC² in Address Search mode where character assembly does not begin until the 8-bit character (secondary address field) following the starting flag of a message matches either the address programmed into CR6 or the global address 11111111.

Receiver CRC Enable (D_3)

This bit enables and disables (1 = enable) the CRC checker in COP mode to allow you to selectively include or exclude characters from the CRC calculation. The MPSC² features a one-character delay between the receiver shift register and the CRC checker so that the enabling or disabling takes effect with the last charcter transferred from the shift register to the receiver buffer. Therefore, you have one full character time in which to read the character and decide whether it should be included in the CRC calculation.

Enter Hunt Phase (D_4)

Although the MPSC² receiver automatically enters Sync Hunt Phase after a reset, there are times when you may wish to reenter it, such as when you have determined that synchronization has been lost or, in SDLC mode, to ignore the current incoming message. Writing a 1 into this bit at any time after initialization causes the MPSC² to reenter Hunt Phase.

Auto Enables (D_5)

Setting this bit to 1 causes the DCD and CTS inputs to act as enable inputs to the receiver and transmitter, respectively.

Number of Received Bits/Character (D_6-D_7)

This field specifies the number of data bits assembled to make each character.

You may change the value on the fly while a character is being assembled and if the change is made before the new number of bits has been reached, it affects that character. Otherwise the new specifications take effect on the next character received.

Table G-5.6 Received Bits/Character

D_7	D_6	BITS/CHARACTER
0	0	5
0	1	7
1	0	6
1	1	8

G-5.1.6 Control Register 4

Figure G-5.6 Control Register 4

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
CLOCK RATE		SYNC MODE		NUMBER OF STOP BITS SYNC MODE		PARITY EVEN/ODD	PARITY ENABLE

Parity Enable (D_0)

Setting this bit to 1 adds an extra data bit containing parity information to each transmitted character. Each received character is expected to contain this extra bit and the receiver parity checker is enabled.

Parity Even/Odd (D_1)

Programming a 0 into this bit when parity is enabled causes the transmitted parity bit to take on the value required for odd parity. The received character is checked for odd parity. Conversely, a 1 in this bit signifies even parity generation and checking.

Number of Stop Bits/Sync Mode (D_2-D_3)

This field specifies whether the channel is used in synchronous (or SDLC) mode or in asynchronous mode. In asynchronous mode, this field also specifies the number of bit times used as the stop bit length by the transmitter. The receiver always checks for one stop bit.

Table G-5.7 Stop Bits

D_3	D_2	MODE
0	0	SYNCHRONOUS MODES
0	1	ASYNCHRONOUS 1 BIT TIME (1 STOP BIT)
1	0	ASYNCHRONOUS 1½ BIT TIMES (1½ STOP BITS)
1	1	ASYNCHRONOUS 2 BIT TIMES (2 STOP BITS)

Sync Mode (D_4-D_5)

When the Stop Bits/Sync Mode field is programmed for synchronous modes D_2 D_3 = 00), this field specifies the particular synchronous format to be used. This field is ignored in asynchronous mode.

Table G-5.8 Synchronous Formats

D_5	D_4	MODE
0	0	8-BIT INTERNAL SYNCHRONIZATION CHARACTER (MONOSYNC)
0	1	16-BIT INTERNAL SYNCHRONIZATION CHARACTER (BISYNC)
1	0	SDLC
1	1	EXTERNAL SYNCHRONIZATION ($\overline{SYNC}$ PIN BECOMES AN INPUT)

Clock Rate (D_6-D_7)

This field specifies the relationship between the transmitter and receiver clock inputs (TxC, RxC) and the actual data rate at TxD and RxD. When operating in a synchronous mode you must specify a 1x clock rate. In asynchronous modes, any of the rates may be specified, however, with a 1x clock rate the receiver cannot determine the center of the start bit. In this mode, you must externally synchronize the sampling (rising) edge of RxC with the data.

Table G-5.9 Clock Rates

CLOCK RATE 1	CLOCK RATE 2	
D_7	D_6	CLOCK RATE
0	0	CLOCK RATE = 1x DATA RATE
0	1	CLOCK RATE = 16x DATA RATE
1	0	CLOCK RATE = 32x DATA RATE
1	1	CLOCK RATE = 64x DATA RATE

G-5.1.7 Control Register 5

Figure G-5.7 Control Register 5

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DTR	NUMBER OF TRANSMITTED BITS/CHARACTER		SEND BREAK	TRANSMITTER ENABLE	CRC POLYNOMIAL SELECT	$\overline{RTS}$	TRANSMITTER CRC ENABLE

Transmitter CRC Enable (D_0)

A 1 or a 0 enables or disables, respectively, CRC generator calculation. The enable or disable does not take effect until the next character is transferred from the transmitter buffer to the shift register, thus allowing you to include or exclude specific characters from the

CRC calculation. By setting or resetting this bit just before loading the next character, it and subsequent characters are included or excluded from the calculation. If this bit is 0 when the transmitter becomes empty, the MPSC² goes to the Idle Phase, regardless of the state of the Idle/CRC latch.

RTS (D_1)

In synchronous and SDLC modes, setting this bit to 1 causes the RTS pin to go low while a 0 causes it to go high. In asynchronous mode, setting this bit to 0 does not cause RTS to go high until the transmitter is completely empty. This feature facilitates programming the MPSC² for use with asynchronous modems.

CRC Polynomial Select (D_2)

This bit selects the polynomial used by the transmitter and receiver for CRC generation and checking. A 1 selects the CRC-16 polynomial ($x^{16} + x^{15} + x^2 + 1$). A 0 selects the CRC-CCITT Polynomial ($x^{16} + x^{12} + x^5 + 1$). In SDLC mode, you must select CRC-CCITT. You may use either polynomial in other synchronous modes.

Transmitter Enable (D_3)

After a reset, the transmitted data output (TxD) is held high (marking) and the transmitter is disabled until this bit is set.

In asynchronous mode, TxD remains high until data is loaded for transmission.

In synchronous and SDLC modes, the MPSC² automatically enters Idle Phase and sends the programmed sync or flag characters.

When the transmitter is disabled in asynchronous mode, any character currently being sent is completed before TxD returns to the marking state.

If you disable the transmitter during the Data Phase in synchronous mode, the current character is sent, then TxD goes high (marking).

In SDLC mode, the current character is sent, but the marking line following is zero-inserted. That is, the lines goes low for one bit time out of every five.

You should never disable the transmitter during the SDLC Data Phase unless a reset is to follow immediately. In either case, any character in the buffer register is held.

Disabling the transmitter during the CRC Phase causes the remainder of the CRC character to be bit-substituted with sync (or flag). The total number of bits transmitted is correct and TxD goes high after they are sent.

If you disable the transmitter during the Idle Phase, the remainder of sync (flag) character is sent, then TxD goes high.

Send Break (D_4)

Setting this bit to 1 immediately forces the transmitter output (TxD) low (spacing). This function overrides the normal transmitter output and destroys any data being transmitted although the transmitter is still in operation. Resetting this bit releases the transmiter output.

Transmitted Bits/Character (D_5-D_6)

This field controls the number of data bits transmitted in each character. You may change the number of bits/character by rewriting this field just before you load the first character to use the new specification.

Table G-5.10 Transmitted Bits/Character

TRANSMIT BITS PER CHARACTER 1	TRANSMIT BITS PER CHARACTER	
D_6	D_5	BITS/CHARACTER
0	0	5 OR LESS (SEE BELOW)
0	1	7
1	0	6
1	1	8

Normally each character is sent to the MPSC² right-justified and the unused bits are ignored. However, when sending five bits or less the data should be formatted as shown below to inform the MPSC² of the precise number of bits to be sent.

Table G-5.11 Transmitted Bits/Character for 5 Characters and Less

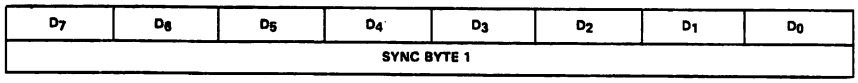
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	NUMBER OF BITS/CHARACTER
1	1	1	1	0	0	0	D_0	1
1	1	1	0	0	0	D_1	D_0	2
1	1	0	0	0	D_2	D_1	D_0	3
1	0	0	0	D_3	D_2	D_1	D_0	4
0	0	0	D_4	D_3	D_2	D_1	D_0	5

DTR (Data Terminal Ready) (D_7)

When this bit is 1, the DTR output is low (active). Conversely, when this bit is 0, DTR is high.

G-5.1.8 Control Register 6

Figure G-5.8 Control Register 6



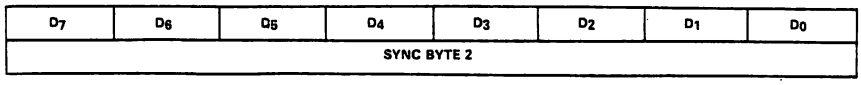
Sync Byte 1 (D_0-D_7)

Sync byte 1 is used in the following modes:

Monosync:	8-bit sync character transmitted during the Idle Phase
Bisync:	Least significant (first) 8 bits of the 16-bit transmit and receive sync character
External Sync:	Sync character transmitted during the Idle Phase
SDLC:	Secondary address value matched to Secondary Address field of the SDLC frame when the MPSC² is in Address Search Mode

G-5.1.9 Control Register 7

Figure G-5.9 Control Register 7



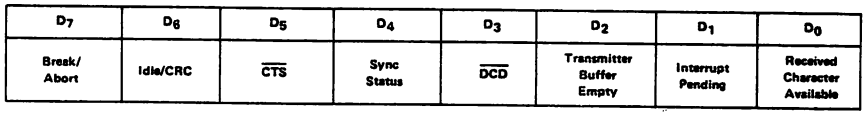
Sync Byte 2 (D_0-D_7)

Sync Byte 2 is used in the following modes:

Monosync:	8-bit sync character matched by the Receiver
Bisync:	Most significant (second) 8 bits of the 16-bit transmit and receive sync characters
SDLC:	You must program the flag character, 01111110, into control register 7 for flag matching by the MPSC² receiver

G-5.1.10 Status Register 0

Figure G-5.10 Status Register 0



D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Break/ Abort	Idle/CRC	$\overline{CTS}$	Sync Status	$\overline{DCD}$	Transmitter Buffer Empty	Interrupt Pending	Received Character Available

Received Character Available (D_0)

When this bit is set, it indicates that one or more characters are available in the receiver buffer for the processor to read. Once all of the available characters have been read, the MPSC² resets this bit until a new character is received.

Interrupt Pending (D_1-Channel A Only)

The interrupt pending bit is used with the interrupt vector register (status register 2) to make it easier to determine the MPSC²'s interrupt status, particularly in Non-vectored Interrupt mode where the processor must poll each device to determine the interrupt source. In this mode, interrupt pending is set when you read status register 2B, the PRI input is active (low) and the MPSC² is requesting interrupt service.

You need not analyze the status registers of both channels to determine if an interrupt is pending. If status affects vector is enabled and interrupt pending is set, the vector you read from SR2 contains valid condition information.

In Vectored Interrupt mode, interrupt pending is set during the interrupt acknowledge cycle (on the leading edge of the 2nd INTA pulse) when the MPSC² is the highest priority device requesting interrupt service (PRI is active). In either mode, if there are no other pending interrupt requests, interrupt pending is reset when the End of Interrupt command is issued.

Transmitter Buffer Empty (D_2)

This bit is set whenever the transmitter buffer is empty, except during the transmission of CRC (the MPSC² uses the buffer to facilitate this function). After a reset, the buffer is considered empty and transmit buffer empty is set.

External/Status Flags

The following status bits reflect the state of the various conditions that cause an external/status interrupt. The MPSC² latches all external/status bits whenever a change occurs that would cause an external/status interrupt (regardless of whether this interrupt is enabled). This allows you to capture transient status changes on these lines with relaxed software timing requirements (see Appendix A for detailed timing specifications).

When you operate the MPSC² in interrupt-driven mode for external/status interrupts, you should read status register 0 when this interrupt occurs and issue a Reset External/Status Interrupt command to reenable the interrupt and the latches. To poll these bits without interrupts, you can issue the Reset External/Status Interrupt command to first update the status to reflect the current values.

DCD (D_3)

This bit reflects the inverted state of the DCD input. When DCD is low, the DCD status bit is high. Any transition on this bit causes an External/Status Interrupt request.

Sync Status (D^4)

The meaning of this bit depends on the operating mode of the $MPSC_2$.

Asynchronous mode: Sync status reflects the inverted state of the SYNC input. When SYNC is low, sync status is high. Any transition on this bit causes an External/Status Interrupt request.

External Synchronization mode: sync status operates in the same manner as asynchronous mode. The MPSC²s receiver synchronization logic is also tied to the sync status bit in external synchronization mode and a low-to-high transition (SYNC input going low) informs the receiver that synchronization has been achieved and character assembly begins (see Appendix A for detailed timing information).

A low-to-high transition on the SYNC input indicates that synchronization has been lost and is reflected both in sync status becoming zero and the generation of an External/Status interrupt. The receiver remains in Receive Data Phase until you set the Enter Hunt Phase bit in Control Register 3.

Monosync, Bisync, SDLC modes: In these modes, sync status indicates whether the MPSC² receiver is in the Sync Hunt or Receive Data Phase of operation. A 0 indicates that the MPSC² is in the Receive Data Phase and a one indicates that the MPSC² is in the Sync Hunt Phase, as after a reset or setting the Enter Sync Hunt Phase bit. As in the other modes, a transition on this bit causes an External/Status interrupt to be issued. You should note that entering Sync Hunt Phase after either a reset or when programmed causes an External/Status Interrupt request which you may clear immediately with a Reset External/Status Interrupt command.

CTS (D_5)

This bit reflects the inverted state of the CTS input. When CTS is low, the CTS status bit is high. Any transition on this bit causes an External/Status Interrupt request.

Idle/CRC (D_6)

This bit indicates the state of the Idle/CRC latch used in synchronous and SDLC modes. After reset this bit is 1, indicating that when the transmitter is completely empty, the MPSC² enters Idle Phase and automatically transmits sync or flag characters.

A zero indicates that the latch has been reset by the Reset Idle/CRC Latch command. When the transmitter is completely empty, the MPSC² sends the 16-bit CRC character and sets the latch again. An External/Status interrupt is issued when the latch is set, indicating that CRC is being sent. No interrupt is issued when the latch is reset.

Break/Abort (D_7)

In asynchronous mode, this bit indicates the detection of a break sequence (a null character plus framing error, that occurs when the RxD input is held low (spacing) for more than 1 character time). Break/Abort is reset when RxD returns high (marking).

In SDLC mode, Break/Abort indicates the detection of an abort sequence when 7 or more ones are received in sequence. It is reset when a zero is received.

Any transition of the Break/Abort bit causes an External/Status Interrupt.

G-5.1.11 Status Register 1

Figure G-5.11 Status Register 1

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
End of SDLC Frame	CRC Framing Error	Overrun Error	Parity Error	SDLC Residue Code			All Sent

All Sent (D_0)

In asynchronous mode, this bit is set when the transmitter is empty and reset when a character is present in the transmitter buffer or shift register. This feature simplifies your modem control software routines. In synchronous and SDLC modes, this bit is always set to 1.

SDLC Residue Code (D_1-D_3)

Since the data portion of an SDLC message can consist of any number of bits and not necessarily an integral number of characters, the MPSC² features special logic to determine and report when the End of Frame flag has been received, the boundary between the data field, and the CRC character in the last few data characters that were just read.

When the end of frame condition is indicated, that is, status register 1 D_7 = 1 and Special Receive Condition interrupt (if enabled), the last bits of the CRC character are in the receiver buffer. The residue code for the frame is valid in the status register 1 byte associated with that data character (remember SR1 tracks the received data in its own buffer).

The meaning of the residue code depends upon the number of bits/characters specified for the receiver. The previous character refers to the last character read before the End of Frame, etc.

Table G-5.12 Residue Codes

8 Bits/Character

D_3	D_2	D_1	Previous Character	2nd Previous Character	
1	0	0	C C C C C C C C	C C C C C D D D	
0	1	0	C C C C C C C C	C C C C D D D D	
1	1	0	C C C C C C C C	C C C D D D D D	
0	0	1	C C C C C C C C	C C D D D D D D	
1	0	1	C C C C C C C C	C D D D D D D D	
0	1	1	C C C C C C C C	D D D D D D D D	(no residue)
1	1	1	C C C C C C C D	D D D D D D D D	
0	0	0	C C C C C C D D	D D D D D D D D	

7 Bits/Character

D_3	D_2	D_1	Previous Character	2nd Previous Character	
1	0	0	C C C C C C C	C C C C C D D	
0	1	0	C C C C C C C	C C C C D D D	
1	1	0	C C C C C C C	C C C D D D D	
0	0	1	C C C C C C C	C C D D D D D	
1	0	1	C C C C C C C	C D D D D D D	
0	1	1	C C C C C C C	D D D D D D D	(no residue)
0	0	0	C C C C C C D	D D D D D D D	

6 Bits/Character

D_3	D_2	D_1	Previous Character	2nd Previous Character	
1	0	0	C C C C C C	C C C C C D	
0	1	0	C C C C C C	C C C C D D	
1	1	0	C C C C C C	C C C D D D	
0	0	1	C C C C C C	C C D D D D	
1	0	1	C C C C C C	C D D D D D	
0	0	0	C C C C C C	D D D D D D	(no residue)

5 Bits/Character

D_3	D_2	D_1	2nd Previous Character	3rd Previous Character	
1	0	0	C C C C C	D D D D D	(no residue)
0	1	0	C C C C D	D D D D D	
1	1	0	C C C D D	D D D D D	
0	0	1	C C D D D	D D D D D	
0	0	0	C D D D D	D D D D D	

Special Receive Condition Flags

The status bits described below (Parity error [if Parity is a Special Receive condition is enabled], Receiver Overrun Error, CRC/Framing Error, and End of SDLC Frame), all represent Special Receive conditions.

When any of these conditions occurs and interrupts are enabled, the MPSC² issues an interrupt request. In addition, if you enabled Condition Affects Vector mode, the vector generated (and the

contents of SR2B for non-vectored interrupts) is different from that of a Received Character Available condition. Thus, you need not analyze SR1 with each character to determine that an error has occurred.

As a further convenience, the Parity Error and Receiver Overrun Error flags are latched, that is, once one of these errors occurs, the flag remains set for all subsequent characters until reset by the Error Reset command. With this facility, you need only read SR1 at the end of a message to determine if either of these errors occurred anywhere in the message. The other flags are not latched and follow each character available in the receiver buffer.

Parity Error (D_4)

This bit is set and latched when parity is enabled and the received parity bit does not match the sense (odd or even) calculated from the data bits.

Receiver Overrun Error (D_5)

This error occurs and is latched when the receiver buffer already contains three characters and a fourth character is completely received, overwriting the last character in the buffer.

CRC/Framing Error (D_6)

In asynchronous mode, a framing error is flagged (but not latched) when no stop bit is detected at the end of a character (i.e. RxD is low 1 bit time after the center of the last data or parity bit). When this condition occurs, the MPSC² waits an additional ½ bit time before sampling again so that the framing error is not interpreted as a new start bit.

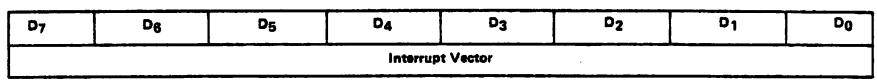
In synchronous and SDLC modes, this bit indicates the result of the comparison between the current CRC result and the appropriate check value and is usually set to 1 since a message rarely indicates a correct CRC result until correctly completed with the CRC check character. Note that a CRC error does not result in a Special Receive Condition interrupt.

End of SDLC Frame (D_7)

This flag is used only in SDLC mode to indicate that the End of Frame flag has been received and that the CRC error flag and residue code is valid. You can reset this flag at any time by issuing an Error Reset command. The MPSC² also automatically resets this bit for you on the first character of the next message frame.

G-5.1.12 Status Register 2

Figure G-5.12 Status Register 2



Interrupt Vector (D_0-D_7 - Channel B Only)

Reading status register 2B returns the interrupt vector that you programmed into control register 2B. If Condition Affects Vector mode is enabled, the value of the vector is modified as follows:

Table G-5.13 Condition Affects Vector Modifications

8085 Modes 8086 Modes	D_4 D_2	D_3 D_1	D_2 D_0	CONDITION
	1	1	1	No Interrupt Pending
	0	0	0	Channel B Transmitter Buffer Empty
	0	0	1	Channel B External/Status Change
	0	1	0	Channel B Received Character Available
	0	1	1	Channel B Special Receive Condition
	1	0	0	Channel A Transmitter Buffer Empty
	1	0	1	Channel A External/Status Change
	1	1	0	Channel A Received Character Available
	1	1	1	Channel A Special Receive Condition

As you can see, code 111 can mean either channel A Special Receive condition or no interrupt pending. You can easily distinguish between the two by examining the Interrupt Pending bit (D_1) of status register 0, channel A. Remember, in Non-vectored Interrupt mode you must read the vector register first for Interrupt Pending to be valid.

G-5.2 MPSC² PROGRAMMING EXAMPLES

```
ASYNC.01
                ********** Asynchronous Mode ***************
Init:
    ISSUE Channel Reset Command (CR0)
    SET Bus Interface Options (CR2A)
    SET Interrupt Vector (CR2B)-if used
    SET Operating Mode (CR4):
      Asynchronous Mode, Parity Select, # of Stop Bits, Clock
      Rate
    SET Receive Enable, Auto Enables, Receive Character Length
    (CR3)
    SET Transmit Enable, Modem Controls, Transmit Char,
    Length (CR5)
    ISSUE Reset External/Status Interrupt Command
    SET Transmit Interrupt Enable, Receive Interrupt on Every
      Character, External Interrupt Enable, Wait Mode Disable.
                    **** End Of Initialization ******

Send:
    ISSUE First Byte To MPSC
    RETURN To Main Program OR Halt

Interrupt:
  CASE Interrupt Type DO:
```

```
Character Received:
     READ Character from MPSC
     PROCESS Character
     ISSUE End Of Interrupt Command
     RETURN From Interrupt

Special Receive Condition:
     READ SR1
     ISSUE Error Reset Command
     CALL Special Error Routine
     ISSUE End Of Interrupt Command
     RETURN From Interrupt

Transmitter Buffer Empty:
     IF Last Character Transferred was End of Message
       THEN ISSUE Reset Transmit Interrupt/DMA Pending
       Command
     ELSE
       Transfer Next Character to MPSC
       ISSUE End Of Interrupt Command
       RETURN From Interrupt

External/Status Change:
     READ SR1
     CALL Special Condition Routine
     ISSUE End Of Interrupt Command
     RETURN From Interrupt

**** END CASE ****

Terminate Transmit:
     RESET Transmit Enable, RTS (CR5)
     RETURN

Terminate Receive:
     RESET Receive Enable (CR1)
     RESET DTR (CR5)

ASYNC.01

     RETURN

END
```

Figure G-5.13 Asynchronous Initialization for Polled Transmit and Receive Receive

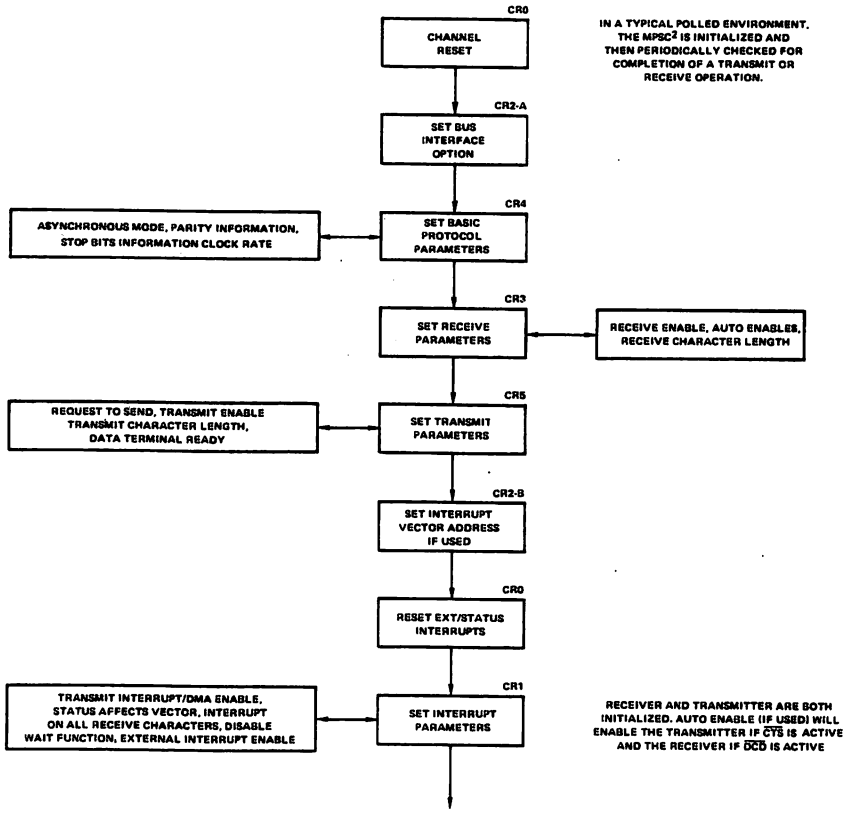


Figure G-5.14 Asynchronous Receive

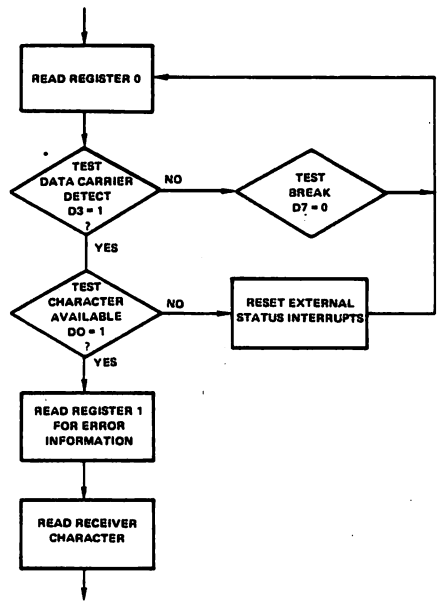
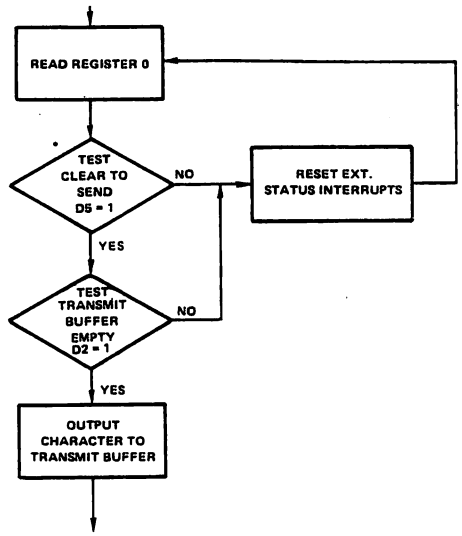


Figure G-5.15 Asynchronous Transmit



*IF AUTO ENABLE WAS SET (D5 = 1 IN CONTROL REGISTER 3), THIS STEP MAY BE OMITTED

SYNC. PRG
**********SYNCHRONOUS OPERATION EXAMPLE**********
****This example uses the Block Transfer Mode***

Init:
    ISSUE Channel Reset Command
    SET Interface Option (CR2A)
    SET Interrupt Vector (CR2B)
    SET Parity Mode, Sync Mode, 1x Clock (CR4)
    SET Sync Character 1 (CR6)
    SET Sync Character 2 (CR7)
    RETURN

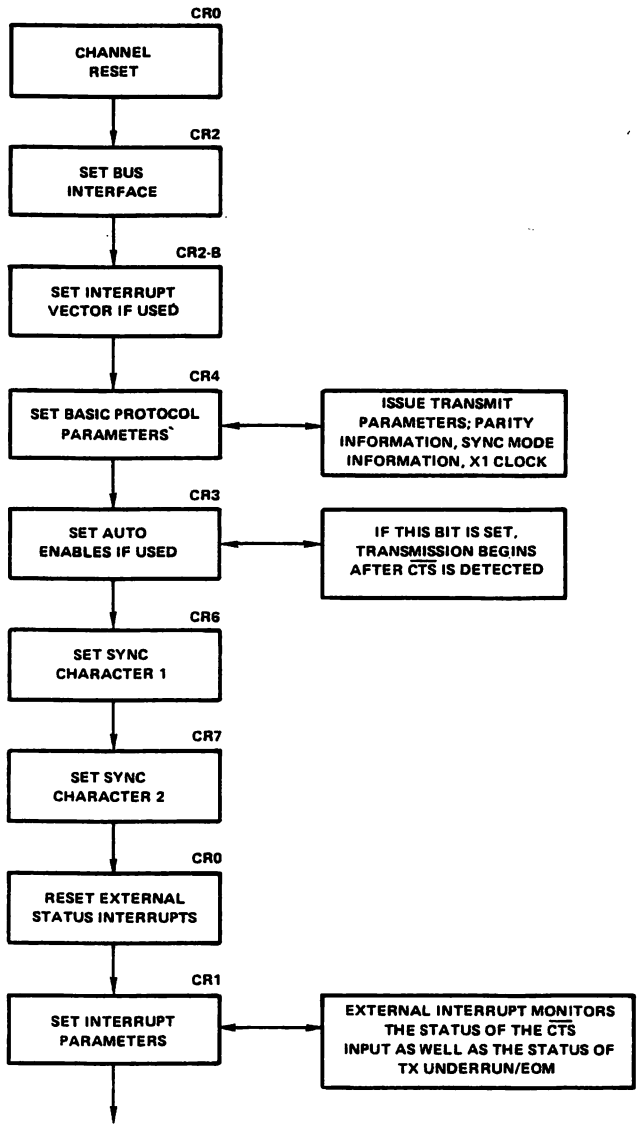
Initiate Transmit:
    ISSUE Reset External/Status Interrupt Command
    SET External Interrupt Enable, Transmit Interrupt Enable
      Wait Enable, Wait on Transmit (CR1)
    SET Transmit Enable, # of Bits/Character, RTS,
      CRC Polynomial Select.

****Transmitter is now enabled and will automatically begin sending Sync characters****

WAIT Several Character Times (a good idea to help system gain synchronization)
Next Message:
    ISSUE Reset Transmit CRC Command

```
Send Character:
    GET Character
    If Character Is To Be Included In CRC
    THEN
      SET CRC Generator On (CR5)
    ELSE
      SET CRC Generator Off (CR5)
    ENDIF
    WRITE Character To MPSC (Processor will "Wait" until
      Transmitter buffer is empty)
    IF Character Was Not The Last
    THEN
      GOTO Send Character (do next character)
    ELSE
      SET CRC Generator On (CR5)
      ISSUE Reset Idle/CRC Latch Command
      WAIT For External/Status Interrupt Indicating CRC Being
      Sent
      IF Next Message Is Ready To Be Transmitted
      THEN
        GOTO Next Message (Next message will be sent
        immediately following CRC)
    ELSE
      WAIT For Transmit Buffer Interrupt indicating Trailing
        Sync Being Sent
      SET Transmitter Enable Off, RTS Off (CR5)
    ENDIF
  ENDIF
  ****End of Transmit Routine****

SYNC.PRG

****Receive Routine****

Receive Message:
    SET External/Status Interrupt Enable, Receive Interrupt
      On First Character Mode, Wait Enabled, Wait on
      Receive (CR1)
    SET Receiver Enable On, Sync Character Load Inhibit,
      # of Bits/Character (CR1)
    SET DTR On (CR5)
    ISSUE Reset External Status Interrupt Command
    ISSUE Enable Interrupt On Next Received Character
      Command
    ISSUE Error Reset Command

    ****Receiver is now enabled and in the Hunt Phase****
    WAIT For External/Status Interrupt (indicating
      synchronization has been achieved)
    Issue Error Reset Command
    WAIT For Received Character Available Interrupt (first
      non-sync character is now available)
    ISSUE Reset CRC Checker Command
    SET Sync Character Load Inhibit Off
```

```
Get Character:
     GET Character from MPSC (processor will "Wait" until at
       least 1 character is available)

     IF Character Is To Be Included In CRC Calculation
     THEN
       Turn CRC Checker On (CR3)
     ELSE
       SET CRC Checker Off (CR3)
     ENDIF
     IF Character Is Part of Message Data
     THEN
       SAVE Character In Memory
     ENDIF
     IF Character Was NOT End Of Message
     THEN
       GOTO READ Character
     ENDIF

*** End Of Message***
     SET CRC Checker On
     READ 2 CRC Characters
     READ 2 Character (these characters may be part of the
       next message but must be read before CRC will be valid)
     READ SR1 (this must be done immediately so that next
       character status will not overwrite)
     IF Parity OR Overrun OR CRC = Error
     THEN
       GOTO Error Processor
     ENDIF

     IF More Messages Are To Be Received
     THEN
       GOTO Get Next Message

SYNC.PRG

     ELSE
       SET DTR Off
       SET Receive Enable Off
     SET External/Status Interrupts Off, Receiver Interrupt
     Mode Disabled (CR1)
     RETURN

     END
       RETURN
```


CR0
CHANNEL RESET
CR2
SET BUS INTERFACE
CR2-B
SET INTERRUPT VECTOR IF USED
CR4
SET BASIC PROTOCOL PARAMETERS
ISSUE TRANSMIT PARAMETERS; PARITY INFORMATION, SYNC MODE INFORMATION, X1 CLOCK
CR3
SET AUTO ENABLES IF USED
IF THIS BIT IS SET, TRANSMISSION BEGINS AFTER CTS IS DETECTED
CR6
SET SYNC CHARACTER 1
CR7
SET SYNC CHARACTER 2
CR0
RESET EXTERNAL STATUS INTERRUPTS
CR1
SET INTERRUPT PARAMETERS
EXTERNAL INTERRUPT MONITORS THE STATUS OF THE CTS INPUT AS WELL AS THE STATUS OF TX UNDERRUN/EOM

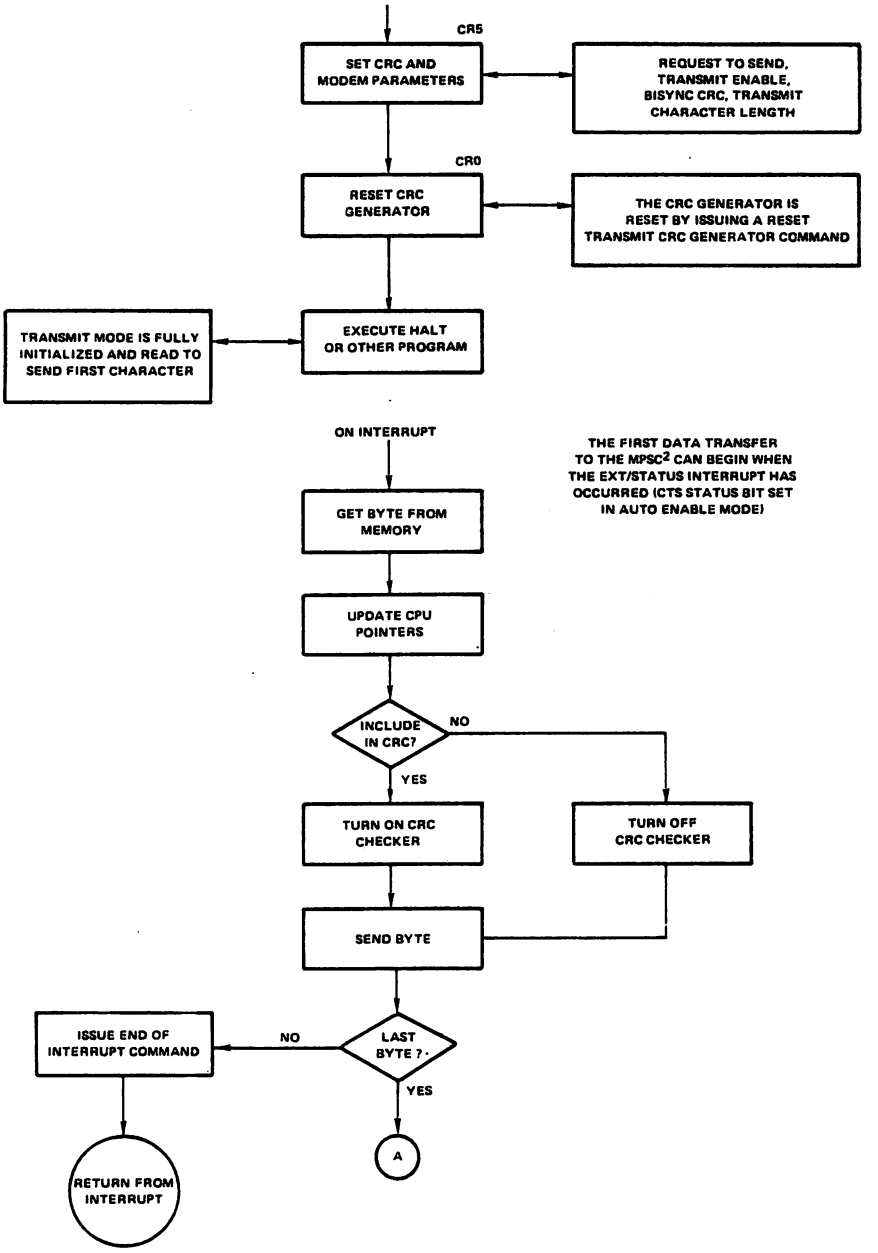
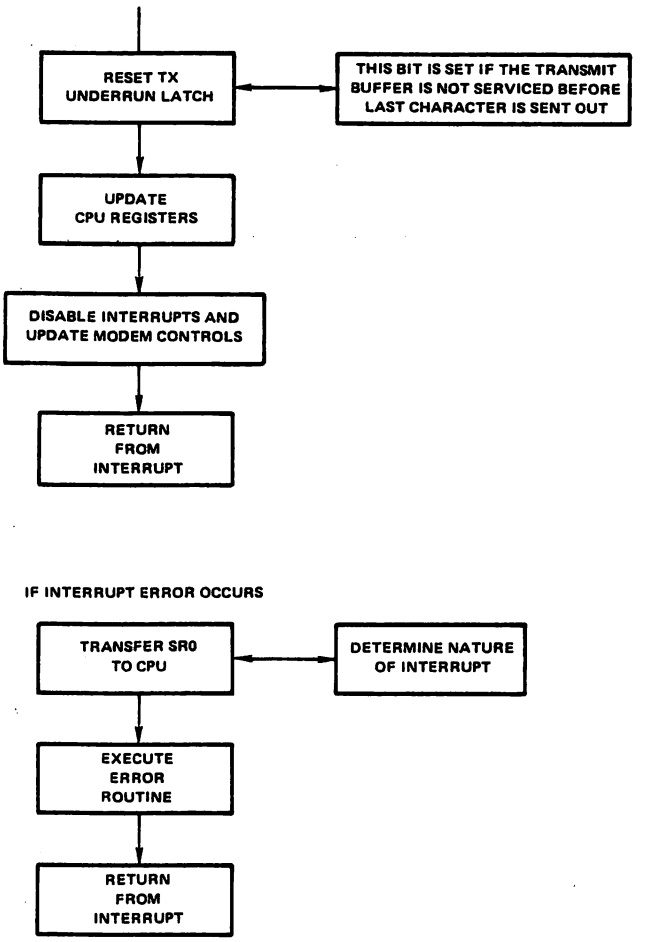
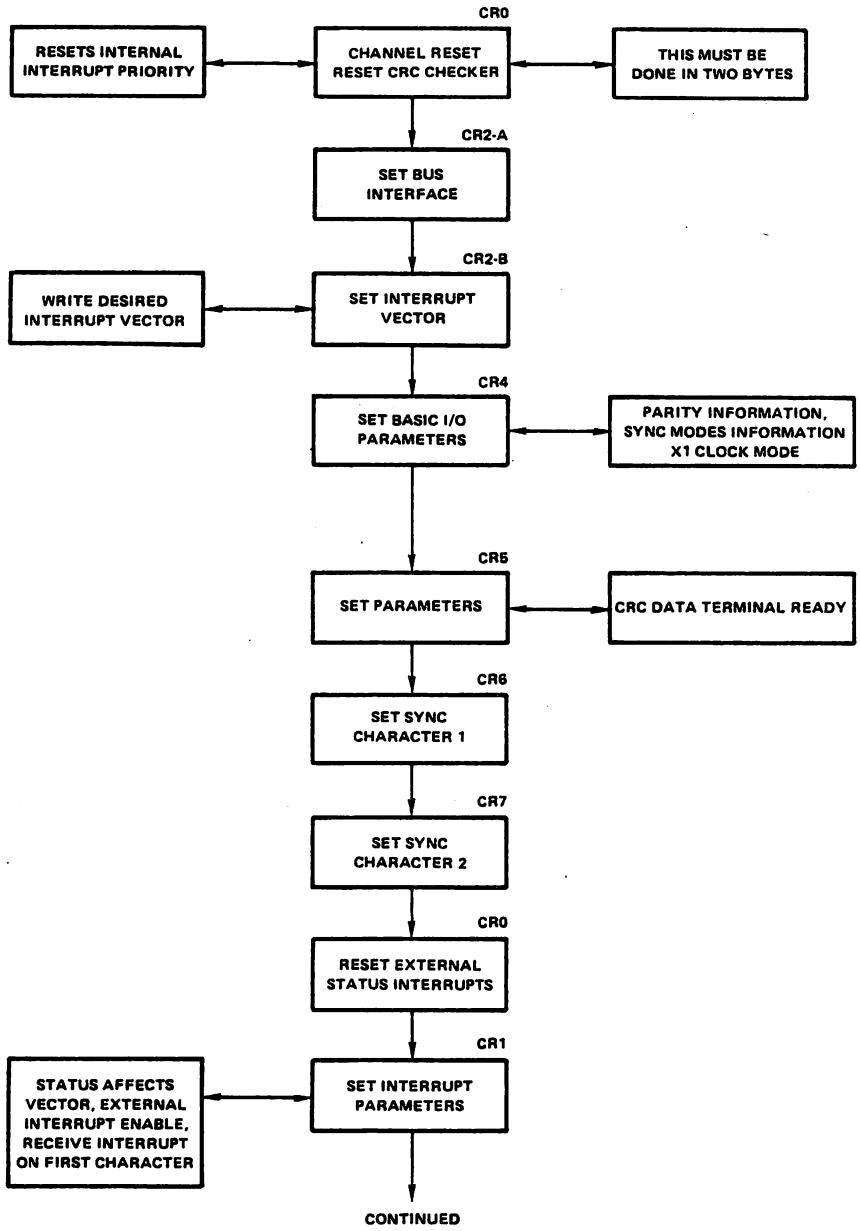

CR5
SET CRC AND MODEM PARAMETERS
REQUEST TO SEND, TRANSMIT ENABLE, BISYNC CRC, TRANSMIT CHARACTER LENGTH
CR0
RESET CRC GENERATOR
THE CRC GENERATOR IS RESET BY ISSUING A RESET TRANSMIT CRC GENERATOR COMMAND
EXECUTE HALT OR OTHER PROGRAM
TRANSMIT MODE IS FULLY INITIALIZED AND READ TO SEND FIRST CHARACTER
ON INTERRUPT
THE FIRST DATA TRANSFER TO THE MPSC2 CAN BEGIN WHEN THE EXT/STATUS INTERRUPT HAS OCCURRED (CTS STATUS BIT SET IN AUTO ENABLE MODE)
GET BYTE FROM MEMORY
UPDATE CPU POINTERS
INCLUDE IN CRC?
NO
YES
TURN ON CRC CHECKER
TURN OFF CRC CHECKER
SEND BYTE
LAST BYTE ?
NO
YES
ISSUE END OF INTERRUPT COMMAND
A
RETURN FROM INTERRUPT

Figure G-5.16 Bisync Initialization Transmit




CR0
RESETS INTERNAL INTERRUPT PRIORITY
CHANNEL RESET RESET CRC CHECKER
THIS MUST BE DONE IN TWO BYTES
CR2-A
SET BUS INTERFACE
CR2-B
WRITE DESIRED INTERRUPT VECTOR
SET INTERRUPT VECTOR
CR4
SET BASIC I/O PARAMETERS
PARITY INFORMATION, SYNC MODES INFORMATION X1 CLOCK MODE
CR5
SET PARAMETERS
CRC DATA TERMINAL READY
CR6
SET SYNC CHARACTER 1
CR7
SET SYNC CHARACTER 2
CR0
RESET EXTERNAL STATUS INTERRUPTS
CR1
STATUS AFFECTS VECTOR, EXTERNAL INTERRUPT ENABLE, RECEIVE INTERRUPT ON FIRST CHARACTER
SET INTERRUPT PARAMETERS
CONTINUED

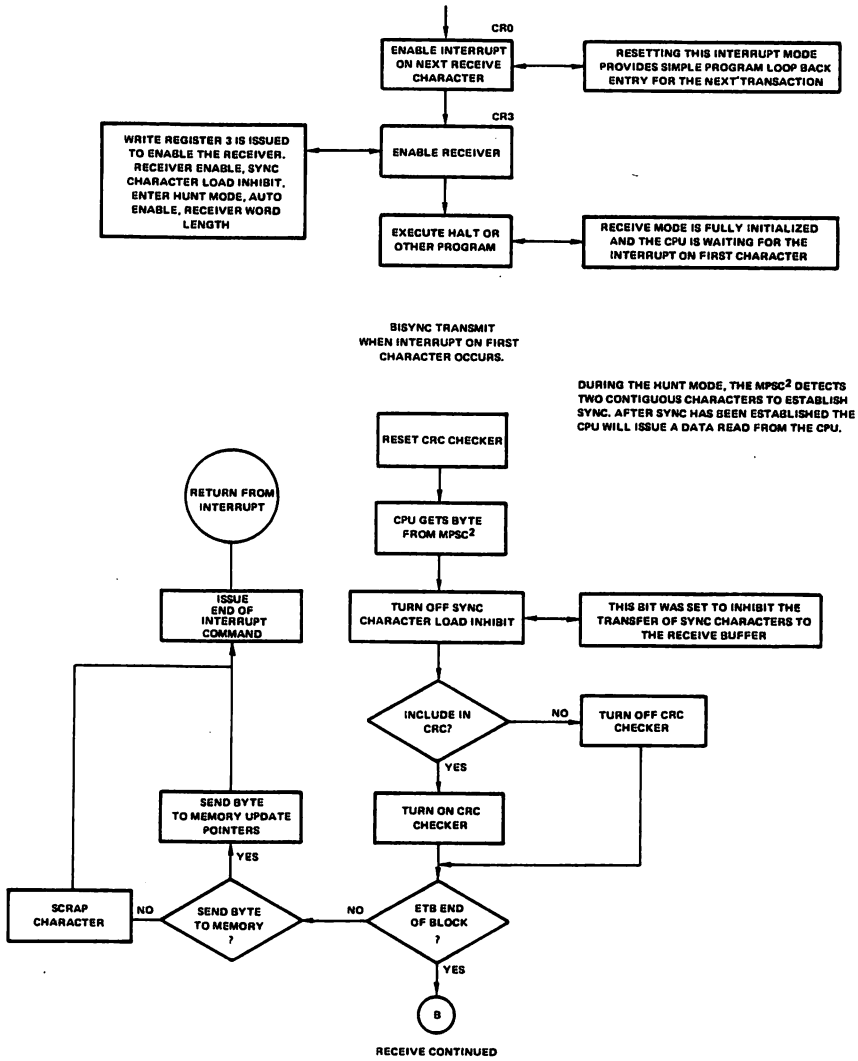
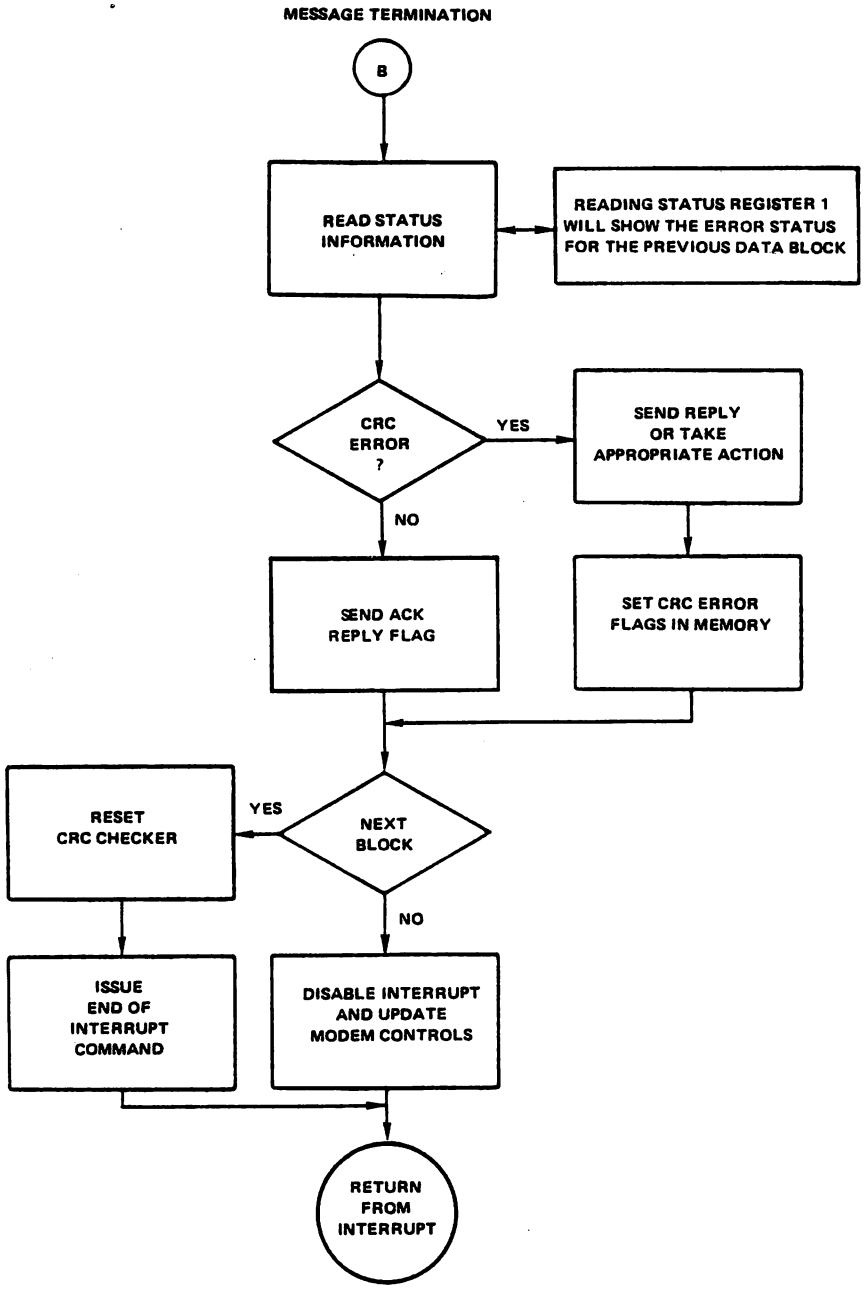

CR0
ENABLE INTERRUPT ON NEXT RECEIVE CHARACTER
RESETTING THIS INTERRUPT MODE PROVIDES SIMPLE PROGRAM LOOP BACK ENTRY FOR THE NEXT TRANSACTION
CR3
WRITE REGISTER 3 IS ISSUED TO ENABLE THE RECEIVER. RECEIVER ENABLE, SYNC CHARACTER LOAD INHIBIT, ENTER HUNT MODE, AUTO ENABLE, RECEIVER WORD LENGTH
ENABLE RECEIVER
EXECUTE HALT OR OTHER PROGRAM
RECEIVE MODE IS FULLY INITIALIZED AND THE CPU IS WAITING FOR THE INTERRUPT ON FIRST CHARACTER
BISYNC TRANSMIT WHEN INTERRUPT ON FIRST CHARACTER OCCURS.
DURING THE HUNT MODE, THE MPSC² DETECTS TWO CONTIGUOUS CHARACTERS TO ESTABLISH SYNC. AFTER SYNC HAS BEEN ESTABLISHED THE CPU WILL ISSUE A DATA READ FROM THE CPU.
RESET CRC CHECKER
RETURN FROM INTERRUPT
CPU GETS BYTE FROM MPSC²
ISSUE END OF INTERRUPT COMMAND
TURN OFF SYNC CHARACTER LOAD INHIBIT
THIS BIT WAS SET TO INHIBIT THE TRANSFER OF SYNC CHARACTERS TO THE RECEIVE BUFFER
INCLUDE IN CRC?
NO
TURN OFF CRC CHECKER
YES
SEND BYTE TO MEMORY UPDATE POINTERS
TURN ON CRC CHECKER
YES
SCRAP CHARACTER
NO
SEND BYTE TO MEMORY ?
NO
ETS END OF BLOCK ?
YES
B
RECEIVE CONTINUED

Figure G-5.17 Bisync Initialization Receive



**********SDLC OPERATION EXAMPLE**********

****This example uses DMA Transfer Mode****

Initialize:
ISSUE Channel Reset Command
SET Interface Option (CR2A)
SET Interrupt Vector (CR2B)
SET SDLC Mode, 1x Clock (CR4)
SET SDLC Flag (CR7)= 01111110
SET SDLC Secondary Address (CR6)
RETURN

Initiate Transmit:
ISSUE Reset External Status Interrupt Command
SET External Interrupt Enable, Transmit Interrupt/DMA Enable (CR1)
SET Transmit Enable, RTS, CRC-CCITT Polynomial (CR5)

****The Transmitter is now enabled and will automatically begin sending Flag characters****

Send Message:
SET DMA Controller to Beginning of Message, # of Characters in Message.
ISSUE Reset Transmit CRC Generator Command
SET 8 Bits/Character (CR5)
WRITE Address byte to MPSC
SET # of Bits/Character (CR5)
ISSUE Reset EOM/CRC Latch Command

****The MPSC will now transmit the message until the DMA Controller completes the required number of transfers****

WAIT for External/Status Change Interrupt (signifies CRC being sent)
IF Next Message Ready to be Transmitted
THEN
  GOTO Send Message (since MPSC will automatically issue a DMA request when ready, set DMA controller to address byte preceding message and skip the write)
ELSE
  ISSUE RESET External/Status Interrupt Command
  ISSUE RESET Transmit Interrupt/DMA Pending Command
  RETURN

****End of Transmit Routine****

Receive Message:
SET External/Status Interrupt Enable, Receive Interrupt on First Character (CR1)
SET Receiver Enable On, 8 Bits/Character, Receive CRC On, Address Search Mode On (CR3)
SET DTR On, CRC-CCITT (CR5)
ISSUE Reset External/Status Interrupt Command
ISSUE Enable Interrupt On Next Character Command

****Receiver is now enabled and in the Hunt Phase****

WAIT for External/Status Interrupt (indicating that a Flag character has been received)
ISSUE Reset External/Status Interrupt Command
RETURN From Interrupt

****Receiver is now in the Address Search Phase****

Next Message:
WAIT for Character Received Interrupt (indicating that an address match or global address has occurred)
GET Address Character (for later processing)
SET DMA Controller
SET # of Bits/Character (CR3)

****Receiver is now in the Data Phase and will transfer all succeeding characters until the End of Frame Flag****

WAIT for Special Receive Condition Interrupt (indicating flag received)
READ SR1 to Obtain CRC Status and Residue Code
SET DMA Controller Off
IF More Messages Are To Be Received
THEN
  GOTO Next Message
ELSE
  SET DTR Off
  SET Receive Enable Off
  RETURN
ENDIF

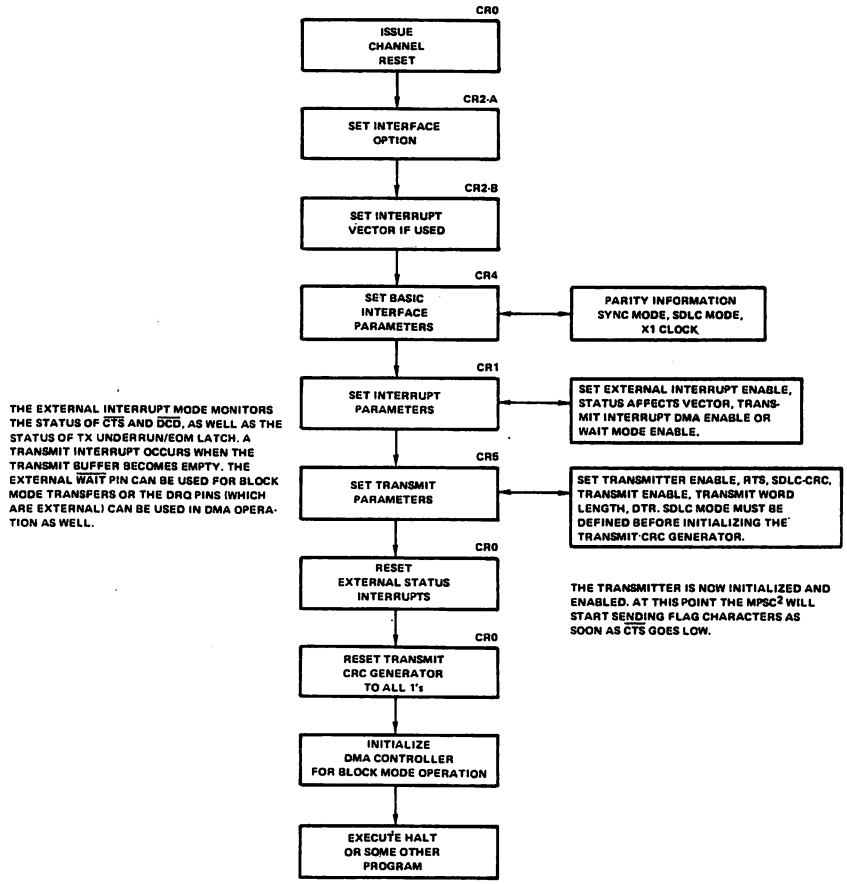
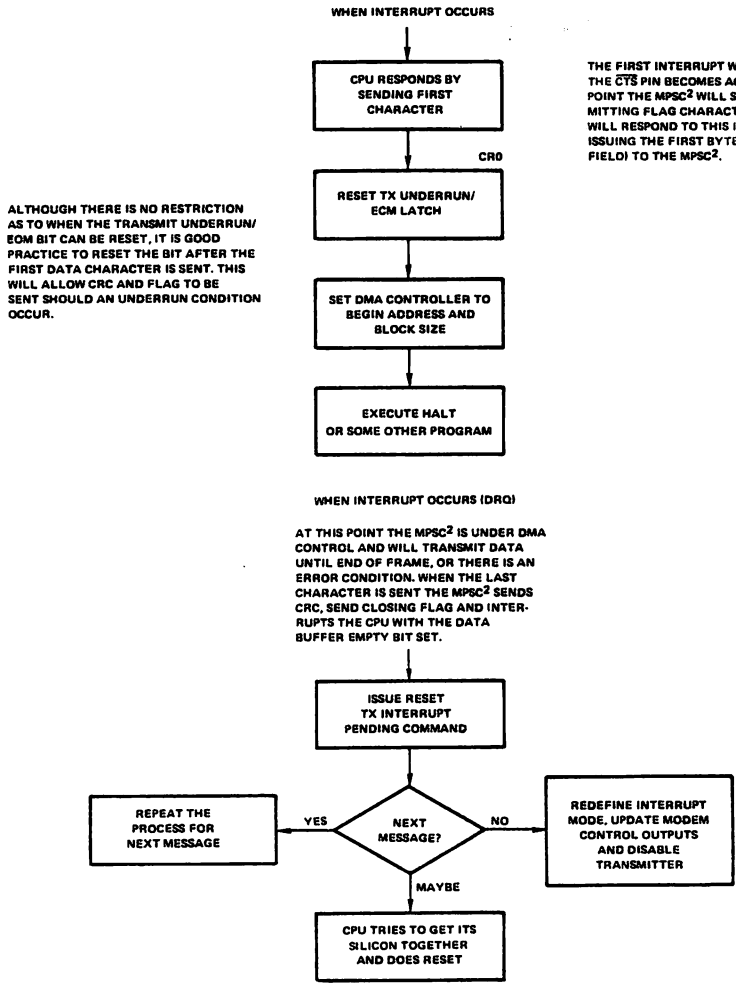
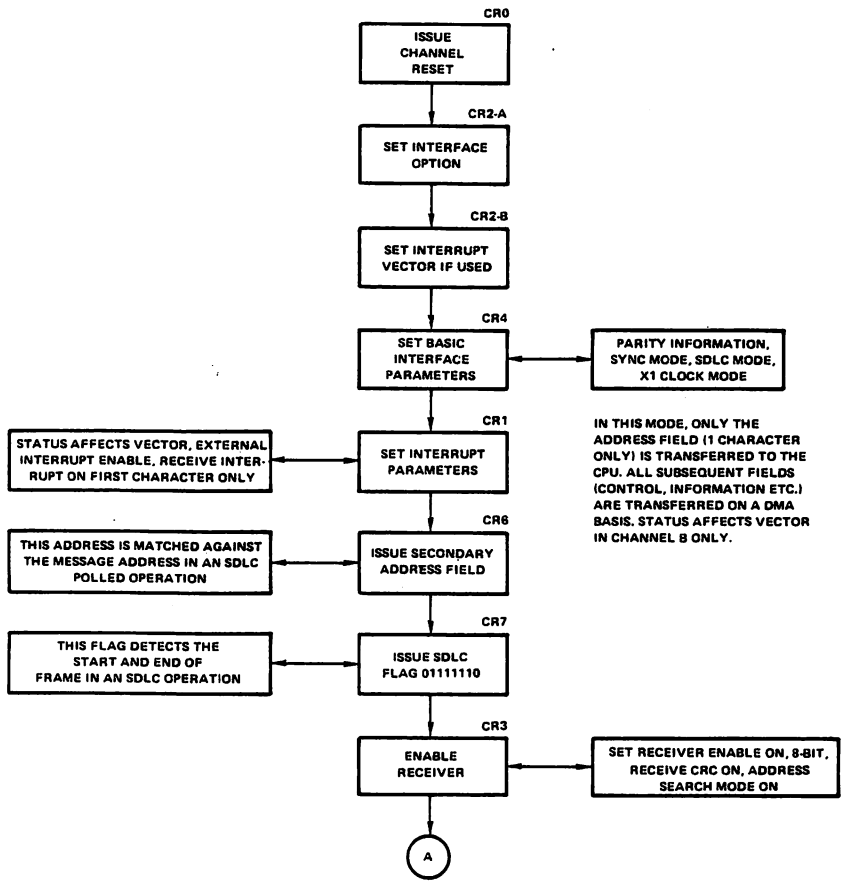
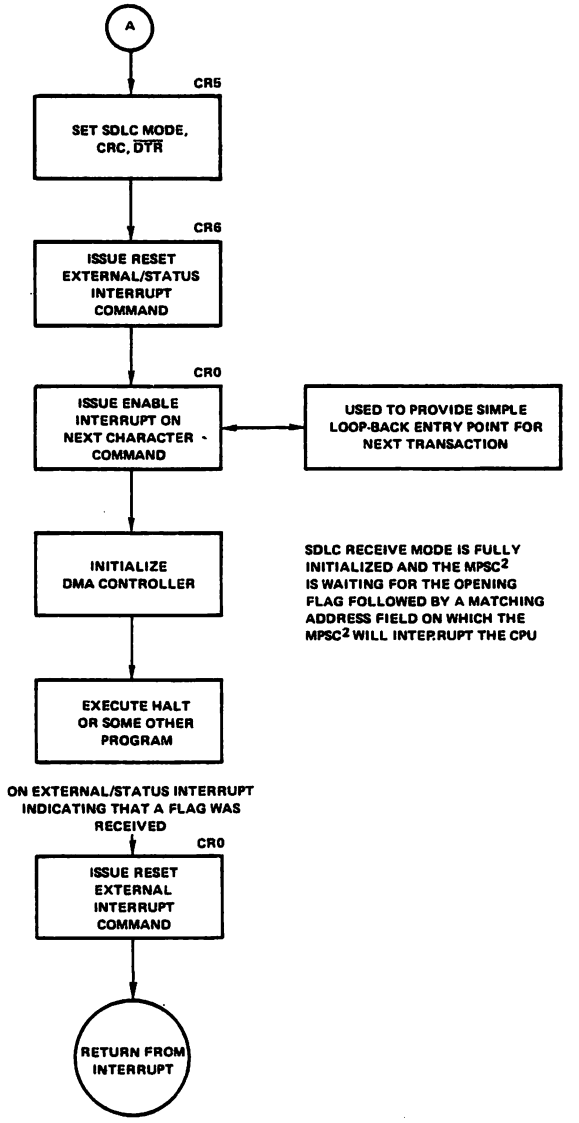

CR0
ISSUE CHANNEL RESET
CR2-A
SET INTERFACE OPTION
CR2-B
SET INTERRUPT VECTOR IF USED
CR4
SET BASIC INTERFACE PARAMETERS
PARITY INFORMATION SYNC MODE, SDLC MODE, X1 CLOCK
CR1
SET INTERRUPT PARAMETERS
SET EXTERNAL INTERRUPT ENABLE, STATUS AFFECTS VECTOR, TRANSMIT INTERRUPT DMA ENABLE OR WAIT MODE ENABLE.
CR5
SET TRANSMIT PARAMETERS
SET TRANSMITTER ENABLE, RTS, SDLC-CRC, TRANSMIT ENABLE, TRANSMIT WORD LENGTH, DTR. SDLC MODE MUST BE DEFINED BEFORE INITIALIZING THE TRANSMIT CRC GENERATOR.
CR0
RESET EXTERNAL STATUS INTERRUPTS
CR0
RESET TRANSMIT CRC GENERATOR TO ALL 1's
INITIALIZE DMA CONTROLLER FOR BLOCK MODE OPERATION
EXECUTE HALT OR SOME OTHER PROGRAM
THE EXTERNAL INTERRUPT MODE MONITORS THE STATUS OF CTS AND DCD, AS WELL AS THE STATUS OF TX UNDERRUN/EOM LATCH. A TRANSMIT INTERRUPT OCCURS WHEN THE TRANSMIT BUFFER BECOMES EMPTY. THE EXTERNAL WAIT PIN CAN BE USED FOR BLOCK MODE TRANSFERS OR THE DRQ PINS (WHICH ARE EXTERNAL) CAN BE USED IN DMA OPERATION AS WELL.
THE TRANSMITTER IS NOW INITIALIZED AND ENABLED. AT THIS POINT THE MPSC2 WILL START SENDING FLAG CHARACTERS AS SOON AS CTS GOES LOW.

Figure G-5.18 SDLC Initialization Transmit




CR0
ISSUE CHANNEL RESET
CR2-A
SET INTERFACE OPTION
CR2-B
SET INTERRUPT VECTOR IF USED
CR4
SET BASIC INTERFACE PARAMETERS
PARITY INFORMATION, SYNC MODE, SDLC MODE, X1 CLOCK MODE
CR1
SET INTERRUPT PARAMETERS
STATUS AFFECTS VECTOR, EXTERNAL INTERRUPT ENABLE, RECEIVE INTERRUPT ON FIRST CHARACTER ONLY
IN THIS MODE, ONLY THE ADDRESS FIELD (1 CHARACTER ONLY) IS TRANSFERRED TO THE CPU. ALL SUBSEQUENT FIELDS (CONTROL, INFORMATION ETC.) ARE TRANSFERRED ON A DMA BASIS. STATUS AFFECTS VECTOR IN CHANNEL B ONLY.
CR6
ISSUE SECONDARY ADDRESS FIELD
THIS ADDRESS IS MATCHED AGAINST THE MESSAGE ADDRESS IN AN SDLC POLLED OPERATION
CR7
ISSUE SDLC FLAG 01111110
THIS FLAG DETECTS THE START AND END OF FRAME IN AN SDLC OPERATION
CR3
ENABLE RECEIVER
SET RECEIVER ENABLE ON, 8-BIT, RECEIVE CRC ON, ADDRESS SEARCH MODE ON
A


A
CR5
SET SDLC MODE, CRC, DTR
CR6
ISSUE RESET EXTERNAL/STATUS INTERRUPT COMMAND
CR0
ISSUE ENABLE INTERRUPT ON NEXT CHARACTER COMMAND
USED TO PROVIDE SIMPLE LOOP-BACK ENTRY POINT FOR NEXT TRANSACTION
INITIALIZE DMA CONTROLLER
SDLC RECEIVE MODE IS FULLY INITIALIZED AND THE MPSC2 IS WAITING FOR THE OPENING FLAG FOLLOWED BY A MATCHING ADDRESS FIELD ON WHICH THE MPSC2 WILL INTERRUPT THE CPU
EXECUTE HALT OR SOME OTHER PROGRAM
ON EXTERNAL/STATUS INTERRUPT INDICATING THAT A FLAG WAS RECEIVED
CR0
ISSUE RESET EXTERNAL INTERRUPT COMMAND
RETURN FROM INTERRUPT

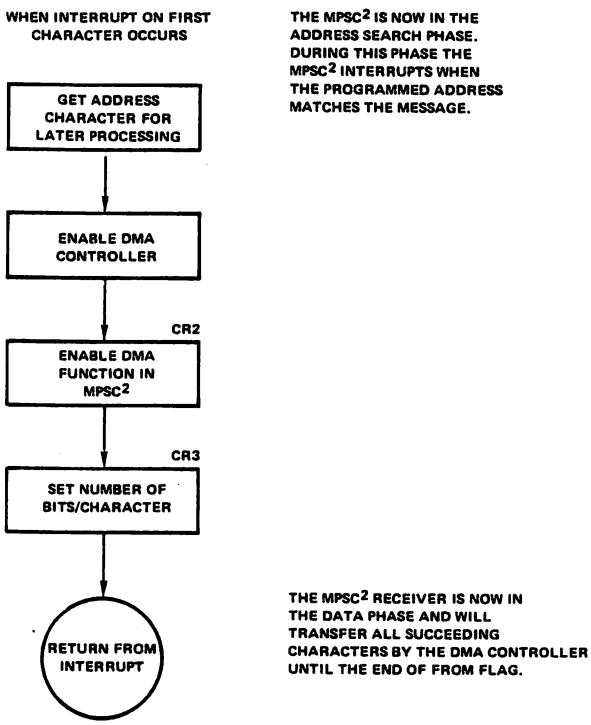
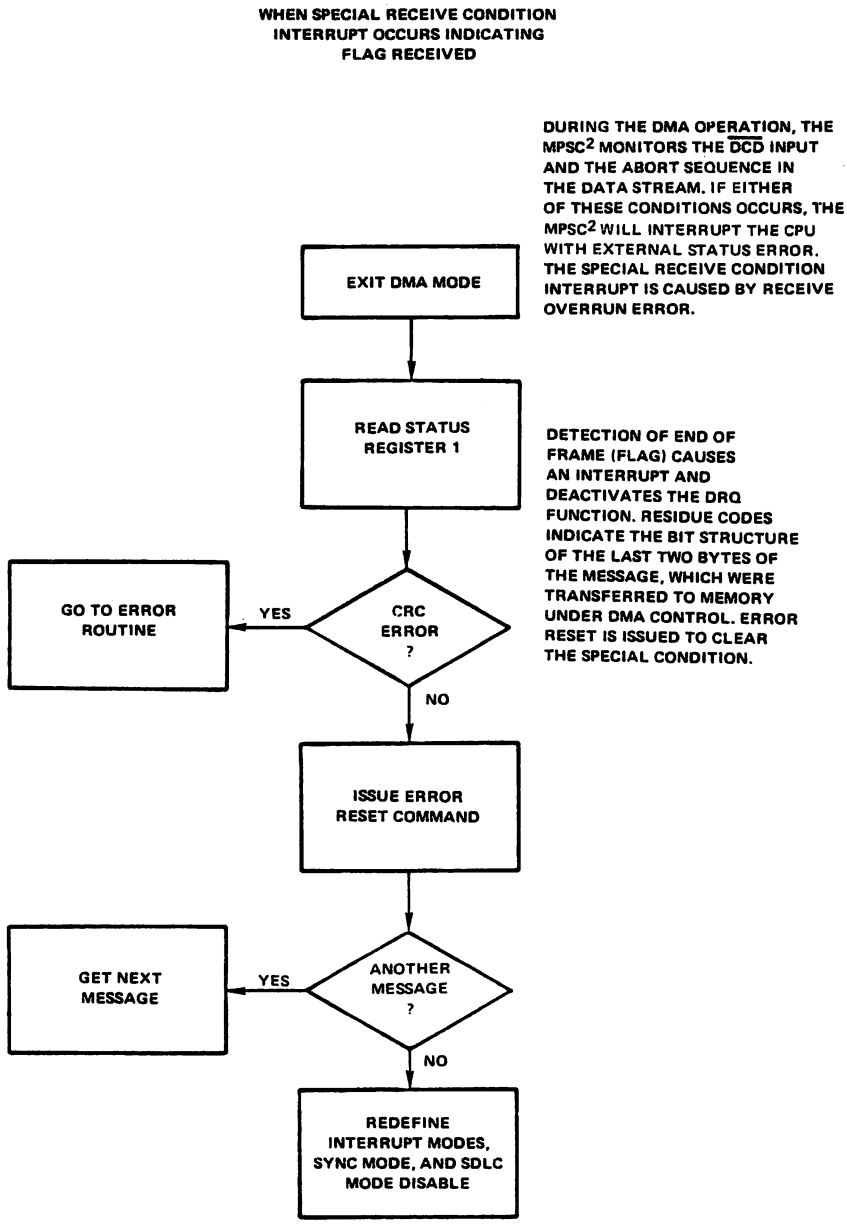

WHEN INTERRUPT ON FIRST CHARACTER OCCURS
GET ADDRESS CHARACTER FOR LATER PROCESSING
ENABLE DMA CONTROLLER
CR2
ENABLE DMA FUNCTION IN MPSC2
CR3
SET NUMBER OF BITS/CHARACTER
RETURN FROM INTERRUPT
THE MPSC2 IS NOW IN THE ADDRESS SEARCH PHASE. DURING THIS PHASE THE MPSC2 INTERRUPTS WHEN THE PROGRAMMED ADDRESS MATCHES THE MESSAGE.
THE MPSC2 RECEIVER IS NOW IN THE DATA PHASE AND WILL TRANSFER ALL SUCCEEDING CHARACTERS BY THE DMA CONTROLLER UNTIL THE END OF FROM FLAG.

Figure G-5.19 SDLC Initialization Receive



G-6 APPLICATION HINTS

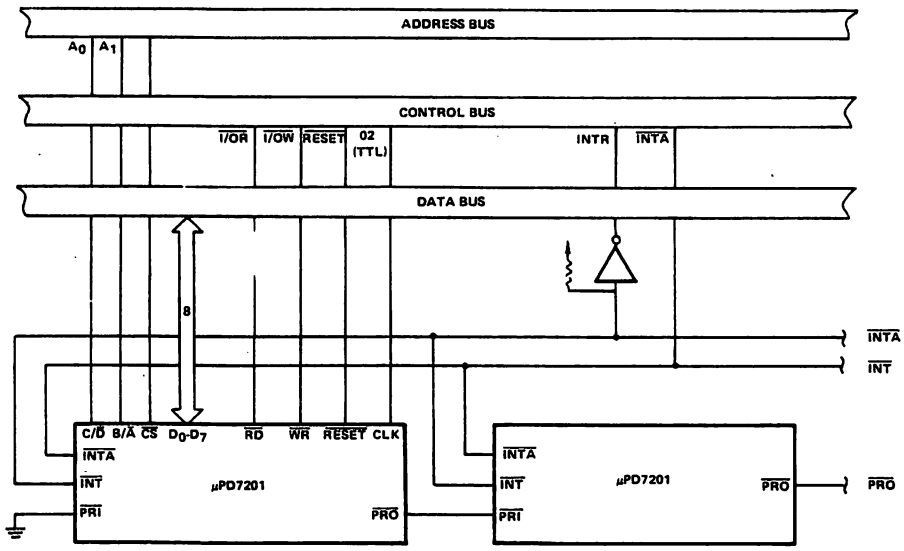
G-6.1 DESIGNING WITH THE MPSC²

Designing the MPSC² into your system is generally straightforward and requires a minimal number of external devices.

G-6.1.1 8080/86-Type Processors

The bus interface used by the MPSC² is directly compatible with 8080/86-type buses. Figure G-6.1 illustrates the basic interconnection scheme for these processors. This configuration supports polled, interrupt driven, and block mode operation.

Figure G-6.1 uPD7201 Interface to 8080 Standard System Bus (Non-DMA)



G-6.1.2 Other Processor Types

You may also connect the MPSC² to uPD780 (Z-80) and 6800/6502-type processors with a few additional gates. Figures G-6.2 and G-6.3, respectively, illustrate the circuits necessary to derive the correct signals. In both cases the MPSC² can be used in Non-vectored mode with minimal software overhead.

Figure G-6.2 uPD780 (Z-80) to MPSC² Adapter

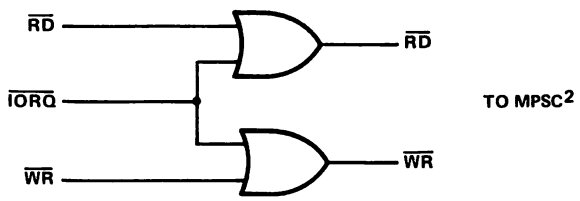
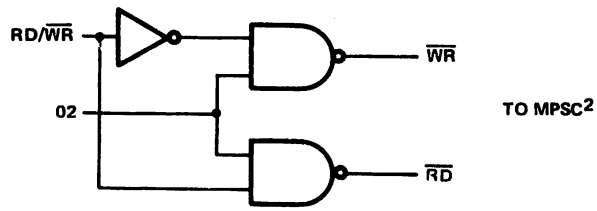
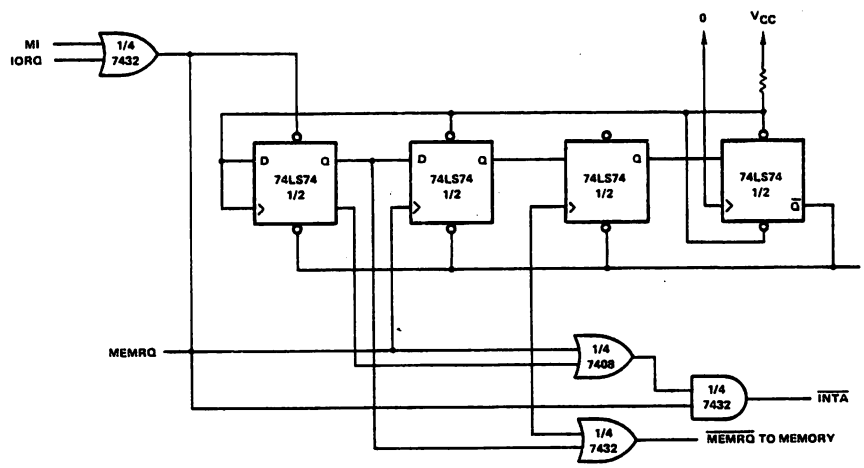


Figure G-6.3 6800/6502 to MPSC² Adapter



The MPSC² can also be used in Vectored Interrupt mode with the uPD780 operated in Interrupt Mode 0. In this mode, the uPD780 handles interrupt requests in much the same manner as an 8080 processor, that is, an interrupt acknowledge sequence is executed during which the processor expects the next instruction to come from the interrupting device. The 8080 INTA signal is generated by combining M1 and IORQ from the uPD780. There is one key difference that must be noted. In accepting a multibyte instruction such as the CALL generated by the MPSC², the 8080 issues a separate INTA pulse for each byte. The uPD780, however, issues an INTA on the first byte only. Succeeding bytes are accessed with memory read cycles. In order for the MPSC² to operate properly, a circuit such as the one shown in Figure G-6.4 should be used to derive the proper INTA sequence.

Figure G-6.4 INTA Generator for Z-80



Most other types of processors may be readily accommodated. The bus control inputs RD, WR, CS, C/D, B/A, and INTA have no timing requirements with respect to the system clock (CLK) and there is no hold time requirement for data after the trailing edge of WR. The only timing constraint you must observe is that the address lines C/D, B/A, and CS must be stable by the leading edge of RD or WR.

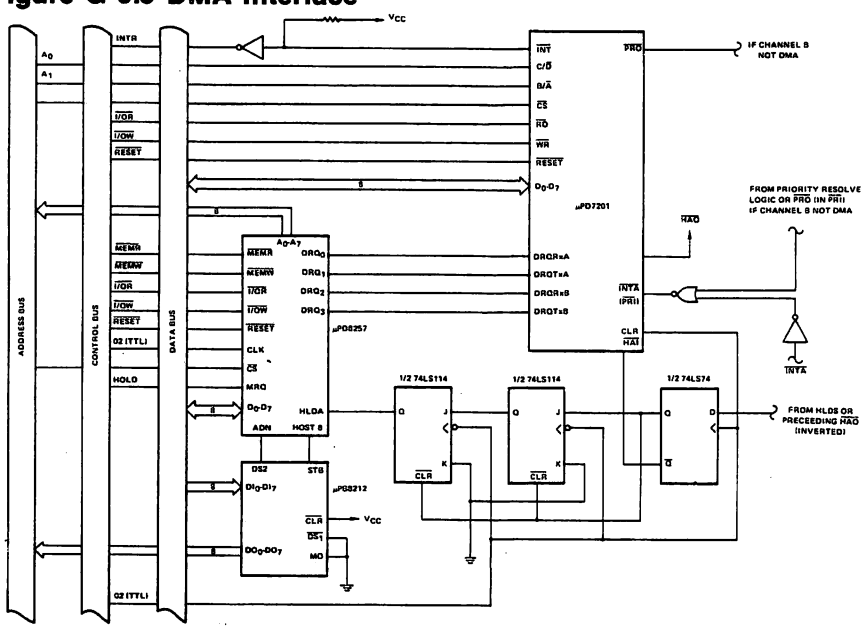
G-6.2 USING THE MPSC² WITH DMA CONTROLLERS

You can greatly increase the data handling capacity of your serial I/O subsystem by using the MPSC² with a DMA controller such as the uPD8257 or uPD8237, to permit direct transfer of data between the MPSC² and memory. Figure 6.5 illustrates a typical MPSC²/DMA configuration. In using the MPSC² in this manner, you should be aware of a few special considerations:

To minimize the number of pins required to implement four DMA channels, the MPSC² does not use the usual DRQ/DACK pins for each channel but rather only DRQ with a single Hold Acknowledge input, HAI. This arrangement eliminates three pins and in addition permits daisychained MPSC²s operating in DMA mode. However, it does require that the MPSC² and the DMA controller reach independent agreement on which DMA request is to be serviced in the case of multiple requests to the same controller.

To ensure that this agreement does occur, you should program the DMA controller for a fixed priority arrangement that agrees with the DMA priority you programmed into the MPSC² (see Secton G-5.1). You must also allow sufficient time for the MPSC² to determine its internal request priority before the DMA controller begins the data transfer. Activating the DMA controller's Hold Acknowledge input through the delay circuit shown in Figure G-6.5 provides this time delay.

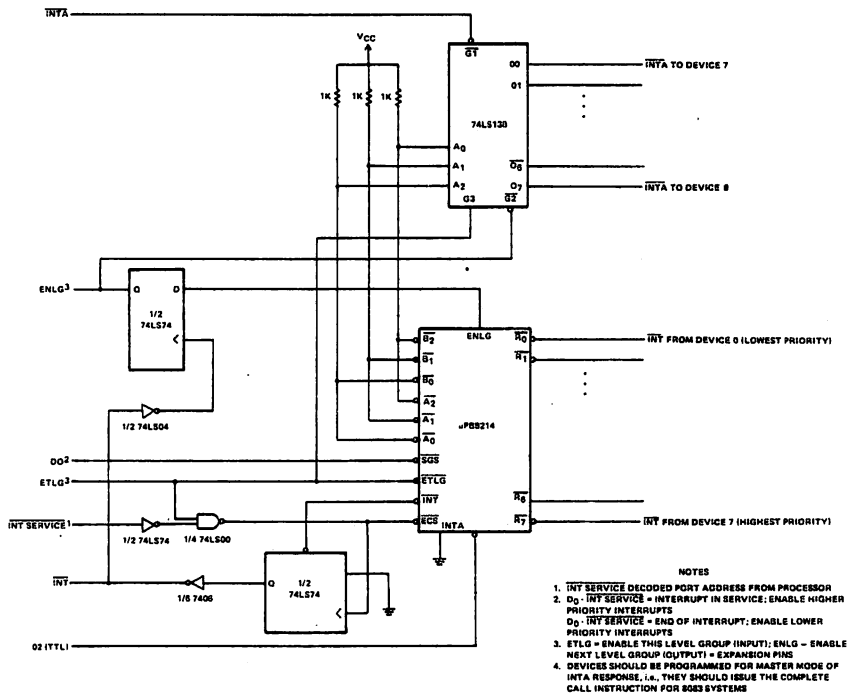
Figure G-6.5 DMA Interface



G-6.3 VECTORED INTERRUPTS WITHOUT USING PRI

There are circumstances when you may wish to use the MPSC²'s Vectored Interrupt feature and you cannot use PRI to inform the MPSC² whether it is the highest priority device requesting service. These situations can occur when both channels are being used in DMA mode (the PRI pin becomes DRQRxB) or when using other peripherals that are incompatible with daisychaining. To retain the Vectored Interrupt feature, you can pull PRI low if available (this is done automatically when both channels are DMA). Program the MPSC² for either 8080 Master or 8086 Vector mode, and gate INTA to the highest priority device with a circuit similar to Figure G-6.6.

Figure G-6.6 Priority Resolution Circuit for Non-daisychained Devices



You should note that an 8259-type interrupt controller programmed for Master Mode does not set its Slave Enable outputs until the second INTA pulse and so is incompatible with the MPSC²'s interrupt acknowledge timing.

G-6.4 TO DMA OR NOT TO DMA...

When operating an MPSC² channel in DMA mode, there are normally some interrupts in parallel with DMA requests. Here are the rules:

Interrupt on Each Character Mode: Both an interrupt and DMA request are made when a character is received.

Interrupt on First Character: The first character received (after issuing an Enable Interrupt On Next Character) generates both an interrupt

and a DMA request. Subsequent characters cause only a DMA request to be issued. As an exception, a Special Receive condition always causes both an interrupt and a DMA request.

Transmitter Buffer Becoming Empty: Only DMA requests are issued when the MPSC² is transmitting under DMA control.

G-6.5 HANDLING AN SDLC UNDERRUN FAULT

Since SDLC-type protocols do not allow flags to be imbedded within a message as filler, a fault condition can sometimes occur where the transmitter runs out of data to send. This situation is particularly common in interrupt-driven systems that are heavily task-loaded. You can use the MPSC²s Idle/CRC latch feature to detect these underrun faults and abort the message before an erroneous End of Frame flag is sent. This is accomplished by issuing a Reset Idle/CRC Latch command to the MPSC² immediately after loading it with the first character of the message. If an underrun condition occurs, the MPSC² automatically begins to send the CRC character calculated up to that point and issues an External/Status Change interrupt to indicate that the CRC is being sent. Since your software routine knows that the end of the message has not been reached, an underrun is indicated and your routine can immediately abort the message with a Send Abort command.

G-6.6 SENDING SYNCHRONOUS PAD CHARACTERS

If you want to send one or more pad characters between synchronous messages, you can do it two ways with the MPSC²:

When the MPSC² issues an External/Status interrupt to indicate that CRC is being sent, you can begin loading your pad characters into the transmitter.

Instead of loading pad characters in response to the above interrupt, you can simply change the programmed sync character on the fly, and the MPSC² will transmit pads when it enters Idle Phase after sending CRC.

G-6.7 TRANSMITTING BISYNC TRANSPARENT MODE

Because of the ability to change the sync registers (CR6, CR7) on the fly, the MPSC² is truly compatible with bisync protocol's Transparent mode. On entering this mode, program CR6 with the DLE character and, if an underrun condition occurs, the correct DLE-SYN sequence is transmitted. On leaving Transparent mode you should reset CR6 back to SYN.

G-6.8 VECTORING THE MPSC² IN NON-VECTORED MODE

If you're using the MPSC² in Non-vectored Interrupt mode, you can still use the Condition Affects Vector feature to direct your software to the correct routine. The following example, written in 8080 assembler, assumes that the MPSC² has been programmed for either 8085 master or slave mode (D_3-D_5 modified) and that CR2B was programmed with a zero.

```
MPSCINT:
        PUSH B          ;Save state so registers are free for
        PUSH D          ;your service routine
        PUSH H
        PUSH PSW
```

```
        MVI A,2         ;Set channel B register pointer to 2
        OUT MPSCBC
        IN MPSCBC       ;Register A = modified vector
        LXI H, JMPTBL   ;HL→ vector jump table
        MVI D,O         ;DE = offset into table
        MOV E,A
        DAD D           ;HL→ jump table + offset
        PCHL            ;Jump to jump table entry

JMPTBL  JMP TBEB        ;Channel B transmitter buffer empty
        NOP
        JMP EXTB        ;External/Status change
        NOP
        JMP RCVB        ;Received character available

        NOP
        JMP SPRB        ;Special receive condition
        NOP
        •
        •              ;Repeat for channel A interrupts
        •
        END
```

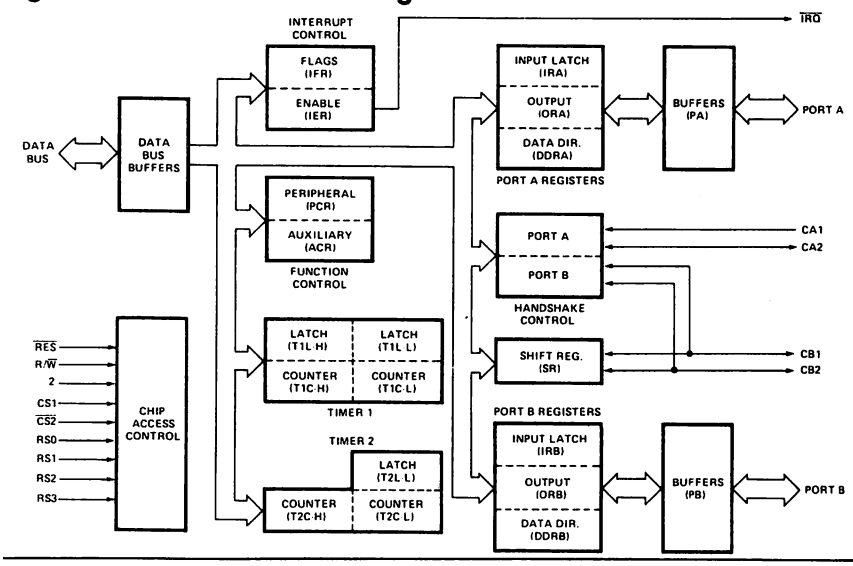
APPENDIX H 6522 VERSATILE INTERFACE SPECIFICATION

- ► Two 8-Bit Bi-directional I/O Ports
- ► Two 16-Bit Programmable Timer/Counters
- ► Serial Data Port
- ► Single +5V Power Supply
- ► TTL Compatible
- ► CMOS Compatible Peripheral Control Lines
- ► Expanded "Handshake" Capability Allows Positive Control of Data Transfers Between Processor and Peripheral Devices
- ► Latched Output and Input Registers
- ► 1 MHz and 2 MHz Operation

The SY6522 Versatile Interface Adapter (VIA) is a very flexible I/O control device. In addition, this device contains a pair of very powerful 16-bit interval timers, a serial-to-parallel/parallel-to-serial shift register and input data latching on the peripheral ports. Expanded handshaking capability allows control of bi-directional data transfers between VIA's in multiple processor systems.

Control of peripheral devices is handled primarily through two 8-bit bi-directional ports. Each line can be programmed as either an input or an output. Several peripheral I/O lines can be controlled directly from the interval timers for generating programmable frequency square waves or for counting externally generated pulses. To facilitate control of the many powerful features of this chip, an interrupt flag register, an interrupt enable register and a pair of function control registers are provided.

Figure H-1: SY6522 Block Diagram



ABSOLUTE MAXIMUM RATINGS

This device contains circuitry to protect the inputs against damage due to high static voltages. However, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages.

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 to +7.0	V
Input Voltage	V_{IN}	-0.3 to +7.0	V
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	-55 to +150	°C

ELECTRICAL CHARACTERISTICS

(V_{CC} 5.0V ±5%, T_A = 0-70° C unless otherwise noted)

Symbol	Characteristic	Min.	Max.	Unit
V_{IH}	Input High Voltage (all except $\phi 2$)	2.4	V_{CC}	V
V_{CH}	Clock High Voltage	2.4	V_{CC}	V
V_{IL}	Input Low Voltage	-0.3	0.4	V
I_{IN}	Input Leakage Current – V_{IN} = 0 to 5 Vdc R/$\overline{W}$, $\overline{RES}$, RS0, RS1, RS2, RS3, CS1, $\overline{CS2}$, CA1, $\Phi 2$	–	±2.5	μA
I_{TSI}	Off-state Input Current – V_{IN} = .4 to 2.4V V_{CC} = Max, D0 to D7	–	±10	μA
I_{IH}	Input High Current – V_{IH} = 2.4V PA0-PA7, CA2, PB0-PB7, CB1, CB2	-100	–	μA
I_{IL}	Input Low Current – V_{IL} = 0.4 Vdc PA0-PA7, CA2, PB0-PB7, CB1, CB2	–	-1.6	mA
V_{OH}	Output High Voltage V_{CC} = min, I_{load} = -100 μAdc PA0-PA7, CA2, PB0-PB7, CB1, CB2	2.4	–	V
V_{OL}	Output Low Voltage V_{CC} = min, I_{load} = 1.6 mAdc	–	0.4	V
I_{OH}	Output High Current (Sourcing) V_{OH} = 2.4V V_{OH} = 1.5V (PB0-PB7)	 -100 -1.0	 – –	 μA mA
I_{OL}	Output Low Current (Sinking) V_{OL} = 0.4 Vdc	1.6	–	mA
I_{OFF}	Output Leakage Current (Off state) $\overline{IRQ}$	–	10	μA
C_{IN}	Input Capacitance – T_A = 25°C, f = 1 MHz (R/$\overline{W}$, $\overline{RES}$, RS0, RS1, RS2, RS3, CS1, $\overline{CS2}$, D0-D7, PA0-PA7, CA1, CA2, PB0-PB7) (CB1, CB2) ($\Phi 2$ Input)	 – – –	 7.0 10 20	 pF pF pF
C_{OUT}	Output Capacitance – T_A = 25°C, f = 1 MHz	–	10	pF
P_D	Power Dissipation	–	700	mW

Figure H-2: Test Load (for all Dynamic Parameters)

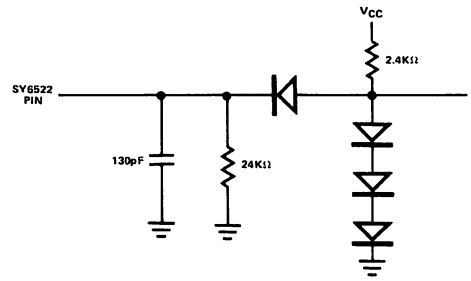
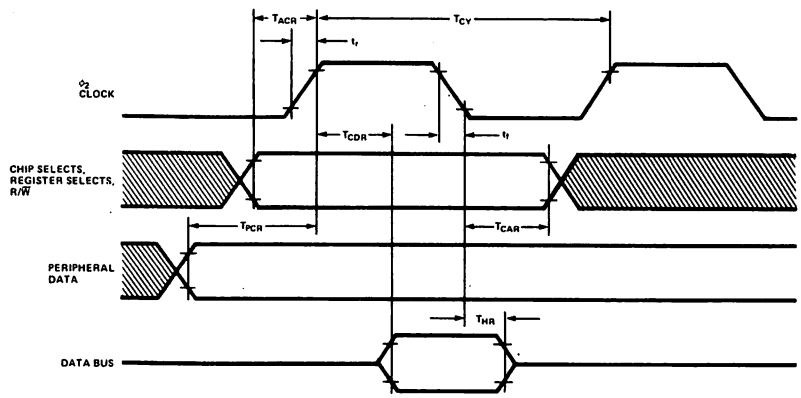


Figure H-3: Read Timing Characteristics

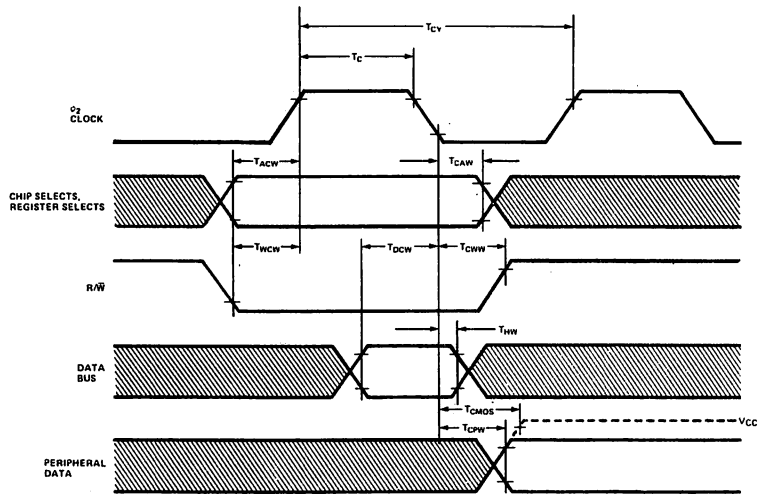


READ TIMING CHARACTERISTICS (FIGURE H-3)

Symbol	Parameter	SY6522		SY6522A		Unit
		Min.	Max.	Min.	Max.	
T_{CY}	Cycle Time	1	50	0.5	50	μs
T_{ACR}	Address Set-Up Time	180	–	90	–	ns
T_{CAR}	Address Hold Time	0	–	0	–	ns
T_{PCR}	Peripheral Data Set-Up Time	300	–	300	–	ns
T_{CDR}	Data Bus Delay Time	–	340	–	200	ns
T_{HR}	Data Bus Hold Time	10	–	10	–	ns

NOTE: tr, tf = 10 to 30ns.

Figure H-4: Write Timing Characteristics



WRITE TIMING CHARACTERISTICS (FIGURE 4)

Symbol	Parameter	SY6522		SY6522A		Unit
		Min.	Max.	Min.	Max.	
T_{CY}	Cycle Time	1	50	0.50	50	μs
T_C	ϕ2 Pulse Width	0.44	25	0.22	25	μs
T_{ACW}	Address Set-Up Time	180	–	90	–	ns
T_{CAW}	Address Hold Time	0	–	0	–	ns
T_{WCW}	R/$\overline{W}$ Set-Up Time	180	–	90	–	ns
T_{CWW}	R/$\overline{W}$ Hold Time	0	–	0	–	ns
T_{DCW}	Data Bus Set-Up Time	300	–	200	–	ns
T_{HW}	Data Bus Hold Time	10	–	10	–	ns
T_{CPW}	Peripheral Data Delay Time	–	1.0	–	1.0	μs
T_{CMOS}	Peripheral Data Delay Time to CMOS Levels	–	2.0	–	2.0	μs

NOTE: tr, tf = 10 to 30ns.

PERIPHERAL INTERFACE CHARACTERISTICS

Symbol	Characteristic	Min.	Max.	Unit	Figure
t_r, t_f	Rise and Fall Time for CA1, CB1, CA2, and CB2 Input Signals	–	1.0	μs	–
T_{CA2}	Delay Time, Clock Negative Transition to CA2 Negative Transition (read handshake or pulse mode)	–	1.0	μs	5a, 5b
T_{RS1}	Delay Time, Clock Negative Transition to CA2 Positive Transition (pulse mode)	–	1.0	μs	5a
T_{RS2}	Delay Time, CA1 Active Transition to CA2 Positive Transition (handshake mode)	–	2.0	μs	5b
T_{WHS}	Delay Time, Clock Positive Transition to CA2 or CB2 Negative Transition (write handshake)	0.05	1.0	μs	5c, 5d
T_{DS}	Delay Time, Peripheral Data Valid to CB2 Negative Transition	0.20	1.5	μs	5c, 5d
T_{RS3}	Delay Time, Clock Positive Transition to CA2 or CB2 Positive Transition (pulse mode)	–	1.0	μs	5c
T_{RS4}	Delay Time, CA1 or CB1 Active Transition to CA2 or CB2 Positive Transition (handshake mode)	–	2.0	μs	5d
T_{21}	Delay Time Required from CA2 Output to CA1 Active Transition (handshake mode)	400	–	ns	5d
T_{IL}	Set-up Time, Peripheral Data Valid to CA1 or CB1 Active Transition (input latching)	300	–	ns	5e
T_{SR1}	Shift-Out Delay Time – Time from ϕ_2 Falling Edge to CB2 Data Out	–	300	ns	5f
T_{SR2}	Shift-In Setup Time – Time from CB2 Data In to ϕ_2 Rising Edge	300	–	ns	5g
T_{SR3}	External Shift Clock (CB1) Setup Time Relative To ϕ_2 Trailing Edge	100	T_{CY}	ns	5g
T_{IPW}	Pulse Width – PB6 Input Pulse	2	–	μs	5i
T_{ICW}	Pulse Width – CB1 Input Clock	2	–	μs	5h
I_{IPS}	Pulse Spacing – PB6 Input Pulse	2	–	μs	5i
I_{ICS}	Pulse Spacing – CB1 Input Pulse	2	–	μs	5h

Figure H-5a: CA2 Timing for Read Handshake, Pulse Mode

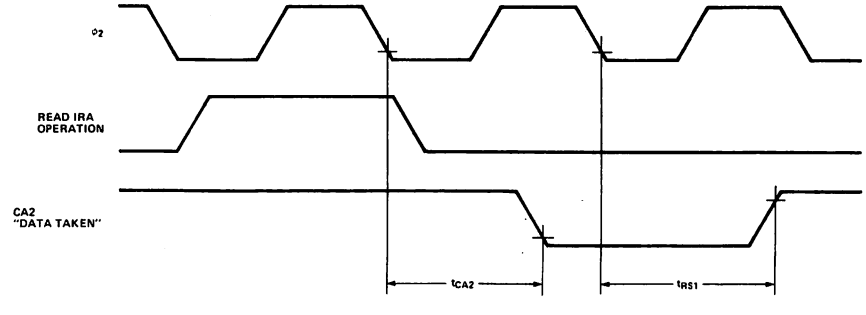


Figure H-5b: CA2 Timing for Read Handshake, Handshake Mode

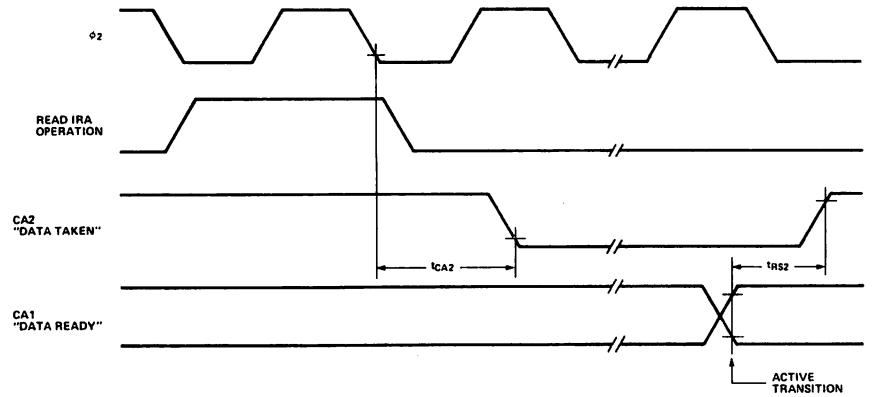


Figure H-5c: CA2, CB2 Timing for Write Handshake, Pulse Mode

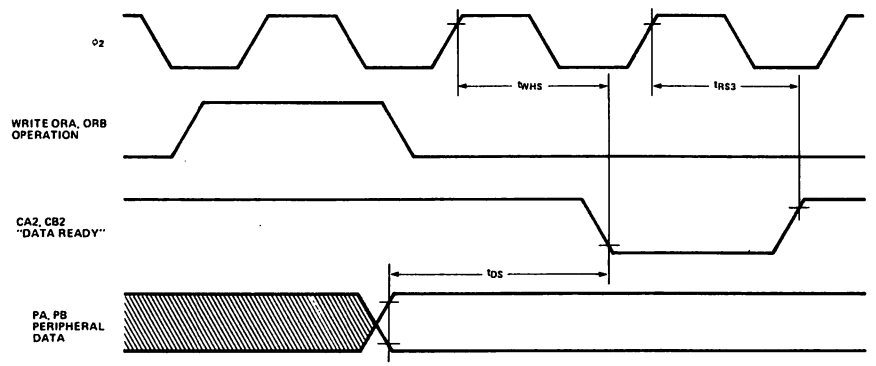


Figure H-5d: CA2, CB2 Timing for Write Handshake, Handshake Mode

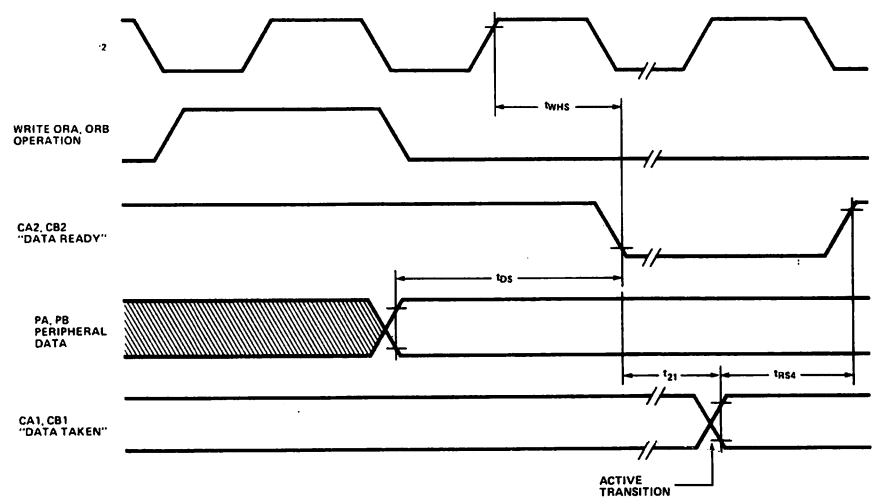


Figure H-5e: Peripheral Data Input Latching Timing

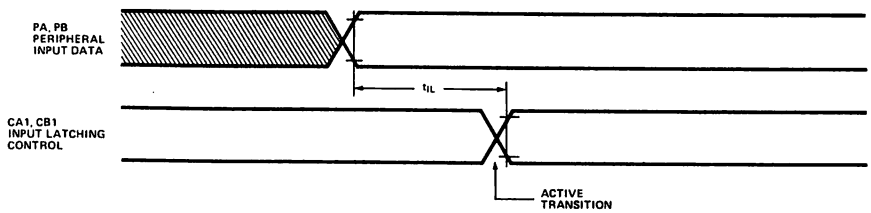


Figure H-5f: Timing for Shift Out with Internal or External Shift Clocking

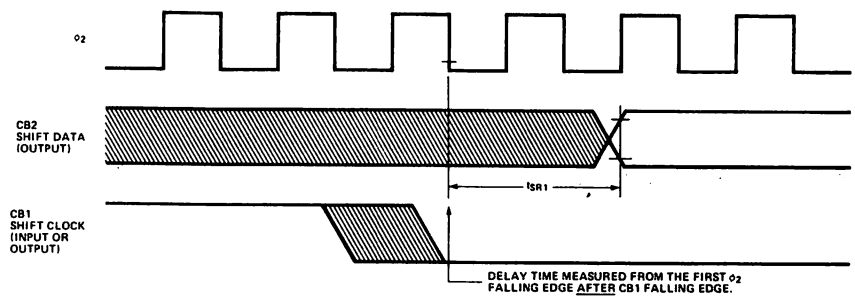


Figure H-5g: Timing for Shift In with Internal or External Shift Clocking

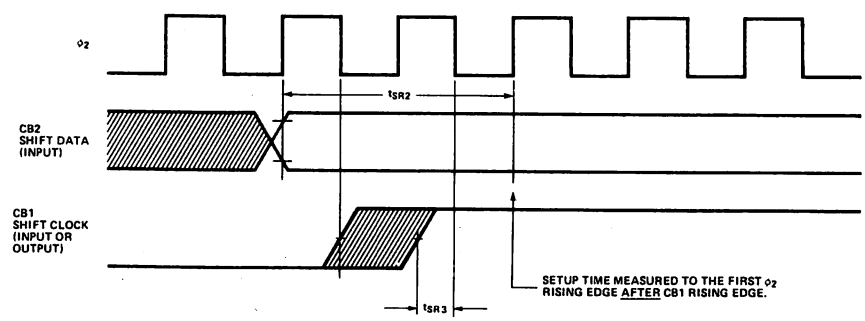


Figure H-5h: External Shift Clock Timing

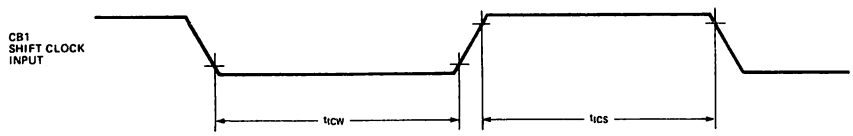
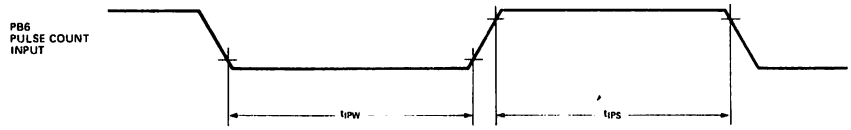


Figure H-5i: Pulse Count Input Timing



PIN DESCRIPTIONS

RES (RESET)

The reset input clears all internal registers to logic 0 (except T1 and T2 latches and counters and the Shift Register). This places all peripheral interface lines in the input state, disables the timers, shift register, etc. and disables interrupting from the chip.

ϕ2 (INPUT CLOCK)

The input clock is the system ϕ2 clock and is used to trigger all data transfers between the system processor and the SY6522.

R/W (READ/WRITE)

The direction of the data transfers between the SY6522 and the system processor is controlled by the R/W line. If R/W is low, data will be transferred out of the processor into the selected SY6522 register (write operation). If R/W is high and the chip is selected, data will be transferred out of the SY6522 (read operation).

DB0-DB7 (DATA BUS)

The eight bi-directional data bus lines are used to transfer data between the SY6522 and the system processor. During read cycles, the contents of the selected SY6522 register are placed on the data bus lines and transferred into the processor. During write cycles, these lines are high-impedance inputs and data is transferred from the processor into the selected register. When the SY6522 is unselected, the data bus lines are high-impedance.

CS1, CS2 (CHIP SELECTS)

The two chip select inputs are normally connected to processor address lines either directly or through decoding. The selected SY6522 register will be accessed when CS1 is high and CS2 is low.

RS0-RS3 (REGISTER SELECTS)

The four Register Select inputs permit the system processor to select one of the 16 internal registers of the SY6522, as shown in Figure H-6.

Figure H-6: SY6522 Internal Register Summary

Register Number	RS Coding				Register Desig.	Description	
	RS3	RS2	RS1	RS0		Write	Read
0	0	0	0	0	ORB/IRB	Output Register "B"	Input Register "B"
1	0	0	0	1	ORA/IRA	Output Register "A"	Input Register "A"
2	0	0	1	0	DDRB	Data Direction Register "B"	
3	0	0	1	1	DDRA	Data Direction Register "A"	
4	0	1	0	0	T1C-L	T1 Low-Order Latches	T1 Low-Order Counter
5	0	1	0	1	T1C-H	T1 High-Order Counter	
6	0	1	1	0	T1L-L	T1 Low-Order Latches	
7	0	1	1	1	T1L-H	T1 High-Order Latches	
8	1	0	0	0	T2C-L	T2 Low-Order Latches	T2 Low-Order Counter
9	1	0	0	1	T2C-H	T2 High-Order Counter	
10	1	0	1	0	SR	Shift Register	
11	1	0	1	1	ACR	Auxiliary Control Register	
12	1	1	0	0	PCR	Peripheral Control Register	
13	1	1	0	1	IFR	Interrupt Flag Register	
14	1	1	1	0	IER	Interrupt Enable Register	
15	1	1	1	1	ORA/IRA	Same as Reg 1 Except No "Handshake"	

IRQ (INTERRUPT REQUEST)

The Interrupt Request output goes low whenever an internal Interrupt Flag is set and the corresponding interrupt enable bit is a logic 1. This output is "open-drain" to allow the interrupt request signal to be "wire-or'ed" with other equivalent signals in the system.

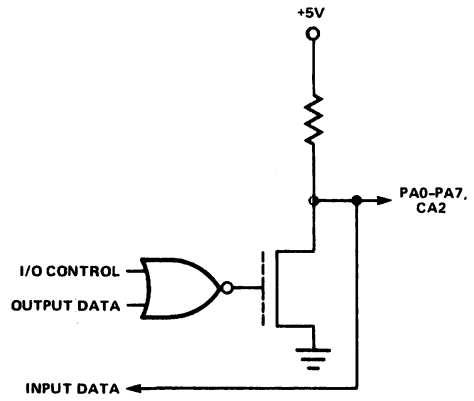
PA0-PA7 (PERIPHERAL A PORT)

The Peripheral A port consists of 8 lines which can be individually programmed to act as inputs or outputs under control of a Data Direction Register. The polarity of output pins is controlled by an Output Register and input data may be latched into an internal register under control of the CA1 line. All of these modes of operation are controlled by the system processor through the internal control registers. These lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. Figure H-7 illustrates the output circuit.

CA1, CA2 (PERIPHERAL A CONTROL LINES)

The two Peripheral A control lines act as interrupt inputs or as handshake outputs. Each line controls an internal Interrupt Flag with a corresponding interrupt enable bit. In addition, CA1 controls the latching of data on Peripheral A port input lines. CA1 is a high-impedance input only while CA2 represents one standard TTL load in the input mode. CA2 will drive one standard TTL load in the output mode.

Figure H-7: Peripheral A Port Output Circuit



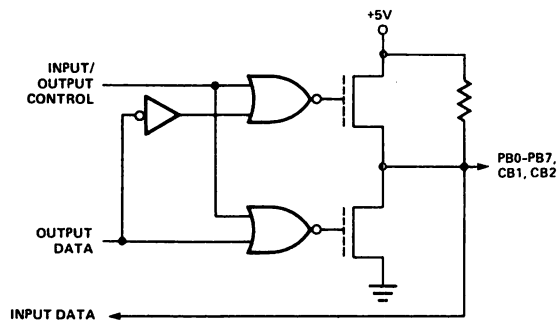
PB0-PB7 (PERIPHERAL B PORT)

The Peripheral B port consists of eight bi-directional lines which are controlled by an output register and a data direction register in much the same manner as the PA port. In addition, the polarity of the PB7 output signal can be controlled by one of the interval timers while the second timer can be programmed to count pulses on the PB6 pin. Peripheral B lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. In addition, they are capable of sourcing 1.0mA at 1.5VDC in the output mode to allow the outputs to directly drive Darlington transistor circuits. Figure H-8 is the circuit schematic.

CB1, CB2 (PERIPHERAL B CONTROL LINES)

The Peripheral B control lines act as interrupt inputs or as handshake outputs. As with CA1 and CA2, each line controls an Interrupt Flag with a corresponding interrupt enable bit. In addition, these lines act as a serial port under control of the Shift Register. These lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. Unlike PB0-PB7, CB1 and CB2 *cannot* drive Darlington transistor circuits.

Figure H-8: Peripheral B Port Output Circuit



FUNCTIONAL DESCRIPTION

PORT A AND PORT B OPERATION

Each 8-bit peripheral port has a Data Direction Register (DDRA, DDRB) for specifying whether the peripheral pins are to act as inputs or outputs. A "0" in a bit of the Data Direction Register causes the corresponding peripheral pin to act as an input. A "1" causes the pin to act as an output.

Each peripheral pin is also controlled by a bit in the Output Register (ORA, ORB) and an Input Register (IRA, IRB). When the pin is programmed as an output, the voltage on the pin is controlled by the corresponding bit of the Output Register. A "1" in the Output Register causes the output to go high, and a "0" causes the output to go low. Data may be written into Output Register bits corresponding to pins which are programmed as inputs. In this case, however, the output signal is unaffected.

Reading a peripheral port causes the contents of the Input Register (IRA, IRB) to be transferred onto the data bus. With input latching disabled, IRA will always reflect the levels on the PA pins. With input latching enabled, IRA will reflect the levels on the PA pins at the time latching occurred (via CA1).

The IRB register operates similar to the IRA register. However, for pins programmed as outputs there is a difference. When reading IRA, the *level on the pin* determines whether a "0" or a "1" is sensed. When reading IRB, however, the bit stored in the *output register*, ORB, is the bit sensed. Thus, for outputs which have large loading effects and which pull an output "1" down or which pull an output "0" up, reading IRA may result in reading a "0" when a "1" was actually programmed, and reading a "1" when a "0" was programmed. Reading IRB, on the other hand, will read the "1" or "0" level actually programmed, no matter what the loading on the pin.

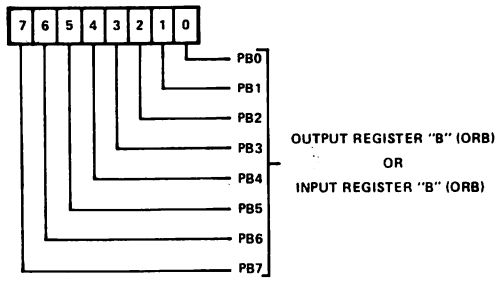
Figures H-9, H-10, and H-11 illustrate the formats of the port registers. In addition, the input latching modes are selected by the Auxiliary Control Register (Figure H-16.)

HANDSHAKE CONTROL OF DATA TRANSFERS

The SY6522 allows positive control of data transfers between the system processor and peripheral devices through the operation of "handshake" lines. Port A lines (CA1, CA2) handshake data on both a read and a write operation while the Port B lines (CB1, CB2) handshake on a write operation only.

Figure H-9: Output Register B (ORB), Input Register B (IRB)

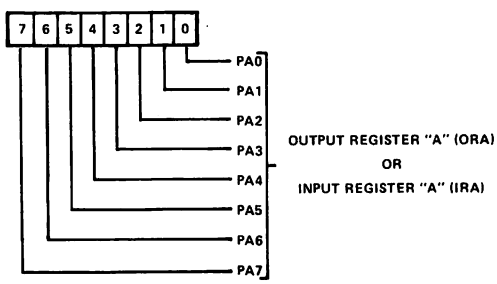
REG 0 – ORB/IRB



Pin Data Direction Selection	WRITE	READ
DDRB = "1" (OUTPUT)	MPU writes Output Level (ORB)	MPU reads output register bit in ORB. Pin level has no affect.
DDRB = "0" (INPUT) (Input latching disabled)	MPU writes into ORB, but no effect on pin level, until DDRB changed.	MPU reads input level on PB pin.
DDRB = "0" (INPUT) (Input latching enabled)		MPU reads IRB bit, which is the level of the PB pin at the time of the last CB1 active transition.

Figure H-10: Output Register A (ORA), Input Register A (IRA)

REG 1 – ORA/IRA



Pin Data Direction Selection	WRITE	READ
DDRA = "1" (OUTPUT) (Input latching disabled)	MPU writes Output Level (ORA).	MPU reads level on PA pin.
DDRA = "1" (OUTPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition.
DDRA = "0" (INPUT) (Input latching disabled)	MPU writes into ORA, but no effect on pin level, until DDRA changed.	MPU reads level on PA pin.
DDRA = "0" (INPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition.